On the logic of unification

Philippe Le Chenadec

INRIA

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél. (1) 39 63 55 11

Rapports de Recherche

N° 1008

Programme 1

ON THE LOGIC OF UNIFICATION

Philippe LE CHENADEC

Mars 1989

RR_1008

On the Logic of Unification

Philippe LE CHENADEC
INRIA B.P. 105 - Rocquencourt
78153 Le Chesnay Cedex France

February 23, 1989

Abstract

Unification, or solving equations on finite trees, is a P-complete problem central to symbolic manipulation, especially in Resolution, Type Inference & Rewriting. We present a natural logic dedicated to unification. This logic enjoys the classical proof-theoretic properties: atomicity; strong normalization; Church-Rosserness; left right, introduction elimination and positive negative symmetries. Motivated by the Type Inference problem, we introduce, besides a model-theoretic semantics and its completeness, a geometrical interpretation of deductions describing their operational content. This allows the design of a normalization process. This unification logic provides significant tools in investigations of higher-order unification, especially for the Type Inference problem, via fixed-point equations deducible from the given equations in Unification Logic. We also present some results on the classification problem of these fixed-point equations. To this end, we introduce the notion of elementary cyclic sets, that essentially possess a single associated fixed-point equation. The finite set of elementary cyclic sets embedded in some unification problem is obtained by a linearization process of the input equations. Finally, up to permutations, there exists a minimum equational deduction associated to an elementary positive occur-check. We give a deterministic algorithm computing this deduction.

Sur la Logique de l'Unification

Résumé

L'unification, ou la résolution d'équations sur les arbres finis, est un problème P-complet que l'on retrouve au cœur des systèmes de calculs symboliques, tels que la Résolution, l'Inférence des Types et la Théorie de la Réécriture. Nous présentons une logique naturelle spécifique à l'unification. Cette logique possède les propriétés classiques que l'on retrouve en Théorie de la Démonstration: atomicité, normalisation forte, propriété de Church-Rosser, symétries gauche droite, introduction elimination et positive négative. Motivé par le problème de l'Inférence des Types, nous introduisons, à coté d'une sémantique style théorie des modèles et sa complétude, une interprétation géométrique des déductions qui décrit leur contenu opératoire. Ceci permet la construction d'une procédure de normalisation. Cette logique de l'unification fournit des outils précieux dans l'étude de l'unification à l'ordre supérieur, et tout particulièrement pour le problème de l'inférence des types, par l'intermédiaire des équations au point fixe déductibles d'un ensemble d'équations donné. A cette fin, nous introduisons la notion d'ensemble cyclique élémentaire, qui possède essentiellement une seule équation au point fixe. L'ensemble fini des ensembles cycliques élémentaires inclus dans un problème d'unification est obtenu par linéarisation des données. Finalement, à permutation près, il existe une déduction équationnelle minimum associée à un ensemble cyclique élémentaire. Nous donnons un algorithme déterministe qui calcule cette déduction.

On the Logic of Unification*

PHILIPPE LE CHENADEC
INRIA, B.P. 105, 78153 Le Chesnay Cedex, France

February 23, 1989

Unification, or solving equations on finite trees, is a P-complete problem central to symbolic manipulation, especially in Resolution, Type Inference & Rewriting. We present a natural logic dedicated to unification. This logic enjoys the classical proof-theoretic properties: atomicity; strong normalization; Church-Rosserness; left right, introduction elimination and positive negative symmetries. Motivated by the Type Inference problem, we introduce, besides a model-theoretic semantics and its completeness, a geometrical interpretation of deductions describing their operational content. This allows the design of a normalization process. This unification logic provides significant tools in investigations of higher-order unification, especially for the Type Inference problem, via fixed-point equations deducible from the given equations in Unification Logic. We also present some results on the classification problem of these fixed-point equations. To this end, we introduce the notion of elementary cyclic sets, that essentially possess a single associated fixed-point equation. The finite set of elementary cyclic sets embedded in some unification problem is obtained by a linearization process of the input equations. Finally, up to permutations, there exists a minimum equational deduction associated to an elementary positive occur-check. We give a deterministic algorithm computing this deduction.

1 Introduction

In proof-theory, since the original work of Gentzen [9] on sequent calculus, much work has been devoted to the normalization process of various logics [11]. Such an analysis was lacking in equational logic (the only exceptions we are aware of are [24,21]). There is a very simple explanation for this oversight: equational logic as traditionally presented lacks

*Part of this research was supported by the Office of Naval Research under contract N00014-84-K-0415 and by the Defense Advanced Research Projects Agency (DOD), ARPA Order No. 5404, monitored by the Office of Naval Research under the same contract. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of DARPA or the U.S. Government.

an elimination rule. However, this elimination rule is omnipresent in Computer Science, disguised under unification, e.g. in Resolution, Rewriting and Type Inference. Unification solves equations between finite trees. It is decidable and fundamental to symbolic manipulation: if there exists a solution there is a most general one, computed by some unification algorithm, cf. §2 below for details. Adding this rule to the traditional introduction rule (Leibniz's principle of substitution of equals for equals) gives us a logic LE with strong properties: atomicity of inferences, strong normalization of deductions, left/right and introduction/elimination symmetries, positive/negative signatures for subexpressions occurrences in deductions.

The study of a logical theory asks for a mathematically interesting interpretation of its theorems. We give two such descriptions for our unification logic. A model-theoretic semantics gives usual completeness. Here the models are (unification) graphs with partial embeddings of terms. Due to the requirement of atomicity, this completeness should not be confused with the algebraic one, à la Birkhoff [15]. The second interpretation is quite new and more penetrating. In Computer Science, the operationality problem for a programming language requires a mathematical description of the computational content of the language. Also, in programming language style, our interpretation gives the correct operational semantics of equational deductions. In geometrical style, it interprets deductions in an algebra $A(\mathcal{E})$ related to the graph of equations $\mathcal{E}$. Such a geometrical semantics allows the design of a syntactical normalization process: we identify two proofs with the same semantics. This strong normalization result is obtained by a *finite* rewriting system via the classical Knuth-Bendix theory [22]. The relevance of this semantics and its operationality are crystal clear when we work at the second-order level, i.e. when the objects become functional. This logic LE possesses deep analogies with Girard's Linear Logic [13]. The discovery of such logics in independant contexts establishes their significance.

Indeed, this logic was coined while working on higher-order equations, in relation with the type inference problem for second-order lambda calculus [12,29], and related systems, such as the Calculus of Constructions [4]. Informally, this problem asks for an algorithm that fills in missing type information in procedures, which lightens the burden of the programmer. It is decidable for very simple type systems, but open for more significant ones. A natural approach to this problem is via higher-order unification, which is undecidable already at order two [14,19]. A typical equation involved in Type Inference has the following form (in Church's λ notation):

$$\lambda\alpha.\Phi(\alpha, X(\alpha)) = \lambda\alpha.F(\Pi(\lambda\beta.\Psi(\alpha, Y(\alpha,\beta))), \Theta(\alpha)),$$

where F is the function space type constructor of type $Type \to (Type \to Type)$, and Π is Church's quantifier of type $(Type \to Type) \to Type$. For readers familiar with second-order lambda calculus, this equation is associated to the typing of the procedure $M = \lambda x.x\ y$ that takes any program x and applies it to the argument y, via the type

structure:

$$\Lambda\alpha.\lambda x^{\Pi(\Phi(\alpha))}.\ \left(x\ X(\alpha)\ (\Lambda\beta.y^{\Pi(\Psi(\alpha))}\ Y(\alpha,\beta))\right)^{\Theta(\alpha)}.$$

The free (functional) variables split into two sets: on one hand we have Φ, Ψ, Θ (subterm types), on the other hand we have X and Y (the type extractions). These two sets are disjoint. Consequently, to these equations is naturally associated a set of first-order equations, obtained by stripping the higher-order structure, here we get $\phi = F(\psi,\theta)$, which is the simple type equation of the procedure M. This is nothing but the familiar idea of first-order approximation of some higher-order complex objects.

If this new set of equations is solvable, say by usual unification, the higher-order equations are trivially solvable. Alternatively, when the associated set $\mathcal{E}$ of first-order equations is not solvable, we want some precise measure of the degree of failure. Ideally, this would allow us to check whether or not the lifting from first to second order overcomes this failure. Due to the origin of the problem, failures originate in the so-called occur-check (or cyclic, recursive, fixed-point equations); and not in the homogeneity test of unification. The desired measure is given by the set of *minimal* deductions of the *minimal* fix-point equations deducible from $\mathcal{E}$ in a restricted version LE_0 of Unification Logic LE, where proper terms of binary inference rules are variables. This set $\mathcal{C}(\mathcal{E})$ gives valuable informations about the higher-order equations. Statements about $\mathcal{C}(\mathcal{E})$ required a normalization procedure for equational deduction. We also introduce a third logical system LE_1, which is a variant of LE_0, well-suited for practical computations arising in the Type Inference problem; and a system LE_2 for equational reasoning in typed λ-calculi.

From these observations, it is quite important to understand the mathematical structure of the set $\mathcal{C}(\mathcal{E})$. A first answer lies in the notion of *elementary cyclic set*. The idea is to distinguish minimal non-unifiable subsets embedded in $\mathcal{E}$. Given $\mathcal{E}$, how should we compute these subsets? Answer: by linear approximations of $\mathcal{E}$. We may non-deterministically and incrementally remove the cycles as follows: if the variable x possesses at least two distinct occurrences in $\mathcal{E}$, one among those can be replaced with a fresh variable. Also a good definition for an "elementary cyclic set" of equations is a set of equations that possesses at least one positive occur-check, but such that removing one equation or linearizing one variable yields a unifiable set of equations. We establish that under this definition there exists a *unique* positive occur-check associated to $\mathcal{E}$. Thus an exhaustive search on $\mathcal{E}$ will give us the finite basis of elementary cyclic sets embedded in $\mathcal{E}$.

Let us see some examples (for formal definitions, cf. §2). The simplest one is $\mathcal{E}_0$: $x = f(x,y)$ (cf. Fig. 1). We have one cycle, the minimum deduction reduces to the hypothesis $\mathcal{E}_0$. Adding $x = f(z,z)$ defines $\mathcal{E}_1$ with two cycles, one primitive, $\mathcal{E}_0$, the other one derived: it owes its existence to the primitive cycle. The example $\mathcal{E}_2$ also defines a unique elementary cycle: $x = f(y,y)$, $x = f(z,f(z,u))$. Here, the minimum deduction

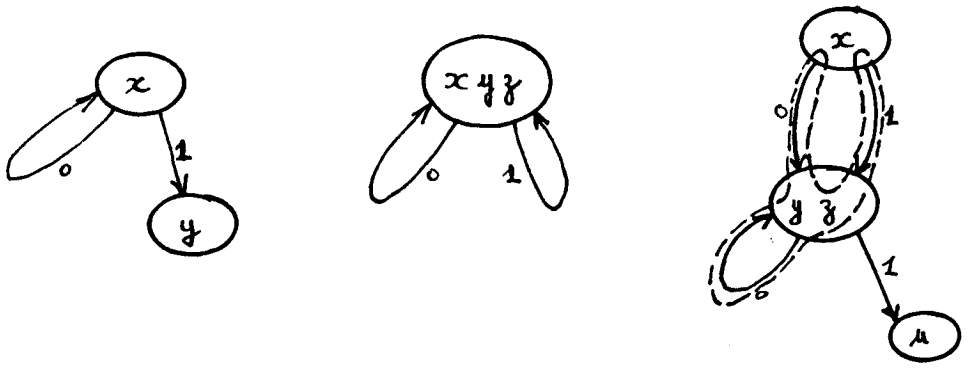

Figure 1: Unification Graphs for $\mathcal{E}_0$, $\mathcal{E}_1$ and $\mathcal{E}_2$

will be:

$$\dfrac{\dfrac{\dfrac{\dfrac{x = f(z, f(z,u))}{f(z, f(z,u)) = x}\,S \quad x = f(y,y)}{f(z, f(z,u)) = f(y,y)}\,T}{z = y}\,E \qquad \dfrac{\dfrac{\dfrac{x = f(y,y)}{f(y,y) = x}\,S \quad x = f(z, f(z,u))}{f(y,y) = f(z, f(z,u))}\,T}{y = f(z,u)}\,E}{z = f(z,u)}\,T$$

To such a deduction is associated a cyclic path in the hypotheses graph that follows the occurrences of the deduction, as pictured in Fig. 1. This path is represented by an expression in the algebra $A(\mathcal{E}_2)$. The deduction possesses an intrinsic, strong, sequentiality: in order to prove the existence of the cycle, you must visit each variable occurrence as specified by the path expression. Each occurrence contributes to the conclusion and none can be omitted. Notice that this basis may include distinct "generators" for the same cycle, e.g. if we add $y = f(y, v)$ to $\mathcal{E}_2$.

Now, in order to solve the fixed point equation $z = f(z,u)$ of the above example, we have two possibilities: either the variable x becomes functional: from the premisses $X(a) = f(y,y)$ and $X(b) = f(z, f(z,u))$ we can no longer deduce the fixed point equation. Or the variable z becomes functional, the deduction is still valid. But its conclusion $Z(a) = f(Z(b), u)$ possesses solutions. Notice that the different arguments feature is essential: the equation $Z(a) = f(Z(a), u)$ has no more solutions than $z = f(z,u)$; and the equational deduction still is valid if we had $X(a) = f(y,y)$ and $X(a) = f(z, f(z,u))$. Naturally, if no conditions are imposed on this lifting process, every set of first-order equations is solvable at second-order. The difficulty comes in by the fact that, in Type Inference, if all variables can become functional, not all occurrences of these variables can receive arguments. From these examples, it is clear that the equational deductions provide adequate tools for expressing the various ways to resolve the fixed points. The principal result here is the existence of a *minimum* deduction associated to an elementary cyclic set. The existence of the minimum deduction follows from both a syntactical reduction via a rewriting process and a semantical analysis of redundancy in deductions. We mention

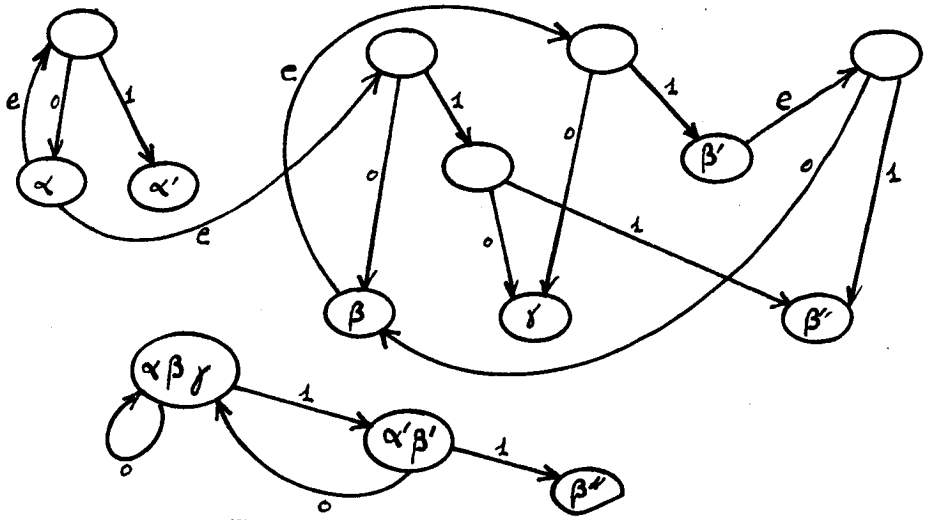

Figure 2: Term and Unification Graph for $\mathcal{E}_M$

another application of the existence of a minimum deduction, of interest for the theory of higher-order unification: to an elementary occur-check is associated a finite automaton [23] recognizing derived sets of equations, where the notion of derivation is borrowed from higher-order unification [19].

Let us see another example from type inference. Consider the following λ-term $M = (\lambda x.xx)\,(\lambda z, y.zyz)$, conjectured to be untypable in second-order λ-calculus by both R. Statman (private communication) and the author. When we try to type this λ-term in simple types [3], we get the following inference tree:

$$\dfrac{\dfrac{(1)\dfrac{x:\alpha \quad x:\alpha}{xx:\alpha'}}{\lambda x.xx:\alpha\to\alpha'} \quad \dfrac{\dfrac{(3)\dfrac{\dfrac{(2)\dfrac{z:\beta \quad y:\gamma}{zy:\beta'}}{zyz:\beta''} \quad z:\beta}{\lambda y.zyz:\gamma\to\beta''}}{\lambda z,y.zyz:\beta\to(\gamma\to\beta'')}}{M:\alpha'}(4)$$

We have to solve on finite trees the equations:

$$\mathcal{E}_M \begin{cases} \alpha = \alpha\to\alpha' & (1)\\ \beta = \gamma\to\beta' & (2)\\ \beta' = \beta\to\beta'' & (3)\\ \alpha = \beta\to(\gamma\to\beta'') & (4)\end{cases}$$

Unification fails (cf. the unification graph in Fig. 2). The set $\mathcal{C}(\mathcal{E}_M)$, among other deductions, contains the three following deductions. Notice that the third deduction is not associated to an elementary cyclic set, the three deductions are expressed in the system

LE_1.

$$\alpha = \alpha \to \alpha' \qquad su\ \frac{\beta' = \beta \to \beta'' \quad \beta = \gamma \to \beta'}{\beta = \gamma \to (\beta \to \beta'')}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{u}{\alpha=\beta\to(\gamma\to\beta'')} \quad \alpha=\alpha\to\alpha'}{\beta=\alpha} \quad \dfrac{t}{\alpha=\beta\to(\gamma\to\beta'')}}{\beta=\beta\to(\gamma\to\beta'')} \quad \beta=\gamma\to\beta'^{\,0}}{\beta'=\gamma\to\beta''} \quad \beta'=\beta\to\beta''}{\gamma=\beta} \quad \beta=\gamma\to\beta'}{\gamma=\gamma\to\beta'}$$

For people having some knowledge of typed λ-calculi, the set $\mathcal{E}_2$ is closely related to the typing in Girard's system F of the λ-term $M = \lambda x, y.\ y\ (x\ I)\ (x\ K)$ [10]. Writing $x : \alpha$, $I = \lambda x : \beta.x$ and $K = \lambda x : \gamma.\lambda y : \delta.x$, typing M in the simply typed λ-calculus fails on the elementary cyclic set $\alpha = (\beta \to \beta) \to \alpha'$, $\alpha = (\gamma \to (\delta \to \gamma)) \to \alpha''$. The two resolutions of the fixed point above gives the two intuitively "minimal" typings of M in F:

$$\lambda x : \forall\alpha.\alpha.\lambda y.\ y \quad (x\ \{(\beta \to \beta) \to \mu\}\ I) \quad (x\ \{(\gamma \to (\delta \to \gamma)) \to \nu\}\ K)$$

$$\lambda x.\lambda y.\ y \quad (x\ \ (\lambda x : \forall\alpha.\alpha.x\ \{\beta \to \forall\alpha.\alpha\})) \quad (x\ \ \lambda x : \forall\alpha.\alpha.\lambda y : \beta.x)$$

The paper is organized as follows. In §2 is established the model-theoretic completeness of our Unification Logic. Especially, we need technical lemmas relating occurrences of variables in the unification graph to occurrences of variables in the input set of equations. The technicalities of this section are quite unavoidable in view of the origin of the problem: strict equations (a variable equals another variable), reputed to be boring ([6], private communication of G. Kahn), are here related to β-redexes. Also we cannot purely ignore them. Moreover these results are important in the following intuitive meaning: the technical propositions describe the local structure of unification graphs. The inference rules of deductions in the logic LE follow this local geometry. To be convinced of this fact, consider the proof search procedure in §5. Also, the detour by technical results is unavoidable for obtaining deep results on Unification Logic. In §3, we introduce the algebra $A(\mathcal{E})$ and establish the strong normalization and Church-Rosser properties for Unification Logic. This is done à la Knuth-Bendix [18,22]. The normal forms enjoy the usual properties of cut-free proofs in natural deduction calculi [11]. We end §3 by applying these proof-theoretic results to Type Inference. A good understanding of this subsection requires familiarity with intuitionistic logic and typed λ-calculi. Especially, we use a form of resolution in minimal predicate calculus. The main result here is a necessary condition for some higher-order equations to possess solutions. Finally, we mention that the normalization results includes a normal form for classical equational logic, and should have applications outside unification theory (e.g. in category theory [28]).

The subsequent parts of the paper are devoted to the analysis of some minimal elements of $\mathcal{C}(\mathcal{E})$, the elementary cyclic sets. These results can be seen as a first application of the analysis of the Unification Logic. In §4, we prove that if a unification problem is non-unifiable in a minimal sense, i.e. if it is an elementary cyclic set, it defines essentially a

unique positive occur-check. While still technical, the arguments are straighforward and proceed by contradiction. A (anonymous) referee kindly pointed out that a simpler proof exists, which is more set-theoretical in spirit. In fact, the comparison of the two proofs is instructive: the "set-theoretic" proof is non-constructive, and its constructivization would yield a proof, but not necessarily the *minimum* one. This is the place to claim that through the paper a strong constructivism approach is taken. If the reader has the feeling that somewhere a simpler proof exists, he should check carefully the details of what is exactly proved. More technically, the second part of Theorem 4.1, establishing this unicity property, cannot be obtained without sweat, and is needed for the (subtle) results of §5. This last section proves the unicity of the minimal deduction of an elementary cyclic set. Here, minimality means both in normal form *and* minimal number of inference steps. We give a deterministic algorithm that finds such a deduction, thus establishing an inherent sequentiality of unification [7,24,33]. Finally, we mention that such results also hold for homogeneity checks. A last word on the structure of the paper: sections 2 and 3 present respectively the model-theory and the proof-theory of the logic LE. They are quite independant from the remaining sections 4 and 5 which are more prospective and technical. The paper is quite self-contained and is readable with a minimal knowledge of typed λ-calculi, rewriting and unification theory and proof theory [3,11,18,22].

2 A Unification Logic and its Completeness

In this section, we present the unification logic in natural deduction style [11] and establish its usual, model-theoretic, completeness.

2.1. Definitions. For clarity, we assume that terms are built up over a binary function symbol f and a denumerable set $\mathcal{V}$ of variables, the general case is a straightforward generalization. The size of a term is its number of occurrences of this binary function symbol. The set of variables occurring in a set of equations $\mathcal{E}$ is noted $\mathcal{V}(\mathcal{E})$. The set of occurrences (binary strings) of the term M is noted $\mathcal{O}(M)$, ϵ being the empty occurrence. The length of occurrence O is noted by $|O|$. For O in $\mathcal{O}(M)$, the subterm of M at occurrence O is noted M/O. We adopt the context notation $C[\_]$ in order to distinguish subterms. The trivial context is noted $[\_]$. The occurrence of the hole is noted O_C. The contexts $C_1[\_]$ and $C_2[\_]$ are equivalent, noted $C_1[\_] \sim C_2[\_]$, iff $O_{C_1} = O_{C_2}$. When O_{C_1} is a prefix of O_{C_2}, we write $C_1[\_] \leq C_2[\_]$ (in the presence of several function symbols, these definitions require that the symbols occurring at the same prefixes of holes occurrences are equal). The composition $C_1[C_2[\_]]$ of contexts $C_1[\_]$ and $C_2[\_]$ will be denoted by juxtaposition $C_1C_2[\_]$. We also need a multiple hole notation $C[\_,\_]$. By convention the occurrence of the first hole is of the form $O0O_1$ while the occurrence of the second one is $O1O_2$.

Let $\mathcal{E}$ be a set of equations between terms. Due to sharing, we assume that equations

are labeled, say by integers, so that two distinct equations can have identical members. An equation is strict when its members are variables. The equivalence relation on $\mathcal{V}(\mathcal{E})$ generated by the strict equations from $\mathcal{E}$ is noted $=_s$. The syntactic equality (identity) of terms or other objects is noted $\equiv$. A solution of $\mathcal{E}$ (on finite trees) is described by a morphism on terms that identifies the members of equations in $\mathcal{E}$. The effective part of such a morphism σ, also called unifier, is finitely generated by the assignment pairs $\sigma(x) = M, x \in \mathcal{V}(\mathcal{E})$. Assume that the set of variables is linearly ordered by $\leq$, and define the greater lower bound of two terms or morphisms by:

- $x \wedge f(M,N) = f(M,N) \wedge x = x$,
- $f(M_1,N_1) \wedge f(M_2,N_2) = f(M_1 \wedge M_2, N_1 \wedge N_2)$,
- $x \wedge y = y \wedge x = x$ if $x \leq y$,
- $(\sigma_1 \wedge \sigma_2)(M) = \sigma_1(M) \wedge \sigma_2(M)$.

The morphism $\sigma_1 \wedge \sigma_2$ is a solution of $\mathcal{E}$ when both σ_1 and σ_2 are. Also, $\bigwedge \sigma$, σ a solution morphism, is the most general unifier of $\mathcal{E}$ (with respect to $\leq$).

In this section, we apply to unification the Curry-Howard [17] isomorphism between Constructive Mathematics and Computer Science (an informal statement that identifies proofs and programs, formulæ and types). According to this paradigm, a unification algorithm, given a set of equations $\mathcal{E}$, computes either a deduction $\vdash x = C[x]$ where $C[x]$ is a non-variable term with a distinguished occurrence of x (i.e. the so-called occur-check is positive, or there exists no finite (integral) solution to $\mathcal{E}$); or a deduction $\vdash f(\overline{M}) = g(\overline{N})$ for distinct function symbols f and g (failure of homogeneity), or a sequence of deductions $\vdash x_i = t_i$, $i = 1,\ldots,n$, defining a most general unifier for $\mathcal{E}$ by assignment pairs. In all cases the deductions proceed from the hypotheses $\mathcal{E}$, according to the logic encoded in the unification algorithm. We analyse the structure of such deductions according to a specific formal system, well-suited for unification. This system made possible the discovery of the geometric interpretation of §3.

This geometry of unification rests on the following graphs. As needed by calculus in the algebra $A(\mathcal{E})$, edges are oriented according to the subterm ordering. A dag is a directed acyclic graph. Firstly, to a set $\mathcal{E}$ of equations we associate its dag representation $G(\mathcal{E})$. This dag has one vertex per variable in $\mathcal{V}(\mathcal{E})$, each member M of an equation from $\mathcal{E}$ defines in the standard way a tree, whose leaves are identified according to their variables giving a term dag $G(M)$ (cf. Fig. 2). Such a dag is rooted. As required by sharing, members of distinct equations are associated to distinct term dags. For each equation $m : M = N$ in $\mathcal{E}$ we have an (oriented) equational edge from the root of $G(M)$ to the root of $G(N)$. Next, the unification graph $U(\mathcal{E})$ is the quotient of $G(\mathcal{E})$ by the smallest, downward closed, equivalence on vertices generated by the equational edges. Finally, we denote by $W(\mathcal{E})$ the quotient graph of $U(\mathcal{E})$ under the smallest upward closed equivalence on vertices. Equivalently, $W(\mathcal{E})$ is the quotient of $G(\mathcal{E})$ by the smallest, downward

closed, *congruence* on vertices generated by the equational edges. The graphs $U(\mathcal{E})$ and $W(\mathcal{E})$ encode in the obvious way a set of terms, usually a proper subset of the set of all terms on $\mathcal{V}(\mathcal{E})$, via graph morphisms or embeddings. Such a morphism between graphs is a pair of functions between vertices and edges respectively compatible with the source-target and labeling (left or right son) structure of edges. Here, we further require that morphisms are compatible with the equivalence classes of terms represented by vertices (i.e. a vertex containing the variable x is mapped to a vertex also containing x). A term can have several embeddings in $U(\mathcal{E})$, but possesses at most one in $W(\mathcal{E})$ (consider $\mathcal{E} = \{1 : x = f(y,z),\ 2 : t = f(y,z)\}$). Usually, such an embedding of M will be unambiguous and the root of $G(M)$ will be denoted $V_U(M)$ or $V_W(M)$, with the subscript dropped if non-ambiguous. Two terms M_1, M_2 are equal modulo $\mathcal{E}$, noted $\models^{\mathcal{E}}_W M_1 = M_2$, iff the terms can be embedded in $W(\mathcal{E})$ so that their roots are equal vertices of $W(\mathcal{E})$. The notation $\models^{\mathcal{E}}_U M_1 = M_2$ means that there exists an embedding of M_1 and M_2 in $U(\mathcal{E})$ with $V_U(M_1) = V_U(M_2)$. Now, a standard unification algorithm computes the graph $U(\mathcal{E})$ and checks its homogeneity (no equivalence class contains two terms with distinct head function symbols) and acyclicity (no directed cycle). As well-known [26], these two conditions characterize unifiability. If so, the most general unifier is easily read on this graph. Intuitively, in §3, we will be interested in multi-paths associated to proofs. They will form a subset of the algebra $A(\mathcal{E})$. However, in the present section, a path will simply be a *directed* path or a pair (v, O), v a vertex and O an occurrence, cf. subsection 2.3 for technical definitions on such paths.

2.2. The Logical System LE. In equational logic, besides the three rules of reflexivity, symmetry and transitivity, we have the rules of replacement and substitution [15]:

$$\frac{M = M' \quad N = N'}{f(M,N) = f(M',N')} \qquad \frac{M = N}{\sigma(M) = \sigma(N)}$$

Obviously these two rules are introduction rules in the sense that they create new terms; but from the view point of the subformula property (objects in the conclusion of some inference should be subobjects of the premisses), they are unsatisfactory, and should be written, if σ on $V(M) \cup V(N)$ is generated by the pairs (x_i, M_i), $i = 1, \ldots, n$:

$$\frac{f(M,N) = f(M,N) \quad M = M' \quad N = N'}{f(M,N) = f(M',N')} \qquad \frac{M = N \quad x_1 = M_1 \cdots x_n = M_n}{\sigma(M) = \sigma(N)}$$

This notation explicits in the premisses every object occurring in the conclusion. Still, this syntax is unsatisfactory as we would like to have a *single* introduction rule. But moving from the first rules to these ones renders them quite similar. Also we drop the requirement that substitutions should substitute only variables, and that replacement should replace only under function symbols. With the requirement of atomicity (one elementary operation

per inference), we get two symmetric rules:

$$IL\ \frac{M = N \quad C[N] = O}{C[M] = O} \qquad IR\ \frac{M = C[N] \quad N = O}{M = C[O]}$$

The equivalence of the two logics is left as an exercice. The mathematically meaningful properties of any logical system follows from the duality introduction-elimination. Turning to the description of unification, we get our elimination rule from the downward closure defining the graph $U(\mathcal{E})$:

$$E\ \frac{C[M] = D[N]}{M = N}$$

provided that the two contexts are equivalent, $C[\_] \sim D[\_]$. Hence our logical system is:

$$LE \left\{ \begin{array}{lll} R\ \dfrac{}{M = M} & S\ \dfrac{M = N}{N = M} & T\ \dfrac{M = N \quad N = O}{M = O} \\ IL\ \dfrac{M = N \quad C[N] = O}{C[M] = O} & E\ \dfrac{C[M] = D[N] \quad C[\_] \sim D[\_]}{M = N} & IR\ \dfrac{M = C[N] \quad N = O}{M = C[O]} \end{array} \right.$$

Practical experience shows that this system leaves little variation on the syntax of its inference rules. We worked for a long time with a system excluding transitivity, the elimination rule being:

$$E\ \frac{C[M] = N \quad N = D[O]}{M = O}$$

where possibly $C[\_] = D[\_] = [\_]$. The reader can prove the equivalence of these systems with respect to the Tarskian semantics. But the semantical reductions of §5 invalidates this last system. Naturally, other, derived, rules may be of interest, e.g.:

$$\frac{CD[M] = E[N] \quad F[N] = GH[O] \quad C[\_] \sim E[\_] \quad F[\_] \sim G[\_] \quad D[\_] \sim H[\_]}{M = O}$$

A word on the meaning of these inference rules. In the model theory tradition, we read: if the premisses are true, the conclusion is true. In the proof theory tradition, we ask what operations are to be performed in order to proceed from the premisses to the conclusion. This interpretation is especially relevant to Computer Science, where research centered around programming language design has faced the operationality problem: find a mathematical, low-level, language, faithful to the high-level programs. Here we propose a geometrical approach to this problem and potspone the exact operational meaning of these rules to the end of §3. For the time being, we record that the binary inference rules just identify or connect two occurrences of a term in the premisses: the subterm N. We call this term the proper term of the inference. In both introduction rules, we distinguish a principal premiss: the left (resp. right) premiss for IL (resp. IR). An auxiliary deduction is a subdeduction whose conclusion is the non-principal premiss of an introduction. Unless

specified, the context of an introduction rule will be non-trivial. An antecedent of some inference rule R is another rule whose conclusion is premiss of the inference R. Remind that in proof-theory, a cut is an introduction immediately followed by the corresponding elimination, the paradigm of computation included in this definition is the cut-elimination process [11].

The investigations recorded in the present paper are motivated by insights in the mathematical structure of the type inference problem. As explained in detail in the introduction, this structure is connected to the set of fixed-point (recursive, cyclic) equations deducible from a simple type inference tree of a procedure. To this end, practical computations with the system LE are necessary. However, only a subsystem LE_0 is useful for fixed-point deductions. The system LE_0 is LE plus the restriction that proper terms are variables. Deductions in LE_0 are sequential, claim justified later on, in §3. The rest of this section is devoted to completeness. Firstly, we establish that the logic LE_0 is complete for path deductions. That is $\vdash^{\mathcal{E}}_{LE_0} x = C[y]$ iff there exists a path $(V(x), O_C)$ in $U(\mathcal{E})$ from $V(x)$ to $V(y)$. This form of completeness is adequate for fixed-point deductions. Completeness for the system LE follows easily. The proof uses quite technical results on the local structure of $U(\mathcal{E})$ and §2.3 can be skipped in a first reading. Why such a technical proof for this apparently shallow result? On one hand, this proof is constructive and suggested the normalization results of §3. On the other hand, the principal argument of the proof rests on a detailed analysis of sharing in unification graphs and data-structure sharing is central to questions of space and time measures in Computer Science. More precisely, in a first step we establish Lemma 2.5 and Proposition 2.6. Their proof is highly technical, but I do not see any way round. They relate the local structure of the graph $U(\mathcal{E})$ to the occurrences of variables in $\mathcal{E}$. In turn, the completeness proof is constructive and explains how a deduction follows the local geometry of the space $U(\mathcal{E})$. A third motivation is given by the Type Inference problem. Here the *form* of the input is significant, and we cannot simply ignore say strict equations (cf. the introduction).

2.3. The Local Structure of the Unification Graph. The present subsection analyses the structure of variable sharing in unification graphs. Also we assume that input equations have the form $x = M$: from the form of the elimination rule, it is clear that this assumption is harmless. To be more formal, there exists an obvious one-to-one correspondance between deductions in LE from $\mathcal{E}$ and deductions in LE from the *simplified* set of equations associated to $\mathcal{E}$. This is *not* true if we consider equivalence classes of variables. First some definitions. A root (resp. leaf) is a vertex with indegree (resp. outdegree) 0. The unification algorithm analyzed here is due to Huet. The input is the set of equations $\mathcal{E}$. The graph $U(\mathcal{E})$ is computed incrementally. To each non-variable subterm of the right-hand sides in $\mathcal{E}$ we associate a unique auxiliary variable from a denumerable set $\mathcal{R}$,

Unification Closure

Input R; V; E;
While $R \neq \emptyset$ do
Select an equation $m : w_1 = w_2 \in R$ with l_m maximal;
$R := R - \{m\}$;
If $\bar{w}_1 \neq \bar{w}_2$
Then
If $suc(\bar{w}_1) = f(w_3, w_4)$ and $suc(\bar{w}_2) = f(w_5, w_6)$
Then $R := R \cup \{m_1 : w_3 = w_5, m_2 : w_4 = w_6\}$;
$l_{m_1} := l_{m_2} := l_m + 1$;
$E := E - \{(0, \bar{w}_2, \bar{w}_5), (1, \bar{w}_2, \bar{w}_6)\}$;
$v := V(w_1) \cup V(w_2)$;
$V := (V - \{V(w_1), V(w_2)\}) \cup \{v\}$;
The representant $\bar{w}$ of v is $\bar{w}_1$ if $suc(\bar{w}_1)$ is defined,
$\bar{w}_2$ otherwise;
Replace $\bar{w}_1$ and $\bar{w}_2$ by $\bar{w}$ in E □

disjoint from $\mathcal{V}$. In an obvious way, this defines two maps:

$$\begin{array}{rcll} suc(w_1) & = & f(w_2, w_3), & w_1 \in \mathcal{R}, w_2, w_3 \in \mathcal{R} \cup \mathcal{V}, \\ suc(w) & = & val(w) = w, & w \in \mathcal{V}, \\ val(w_1) & = & f(val(w_2), val(w_3)), & w_1 \in \mathcal{R},\ suc(w_1) = f(w_2, w_3). \end{array}$$

By this technical trick, we represent each subterm occurrence by a unique variable. The vertex of an $\mathcal{R}$-variable w is noted $V(w)$. An edge is a triple (l, u, v), l its label (0 or 1, left or right son), u its source and v its target. But it will be convenient to represent an edge e from v to v' by a triple $e = (b, w, w')$ with $b = 0$ or 1, $w \in \mathcal{R} \cup \mathcal{V}$, $w \in v$, $w' \in v'$, i.e. equivalence classes are represented by transversal sets.

Define $V = \{\{w\} | w \in \mathcal{V} \cup \mathcal{R}\}$ and $E = \{(0, w, w'), (1, w, w'') \mid suc(w) = f(w', w''), w \in \mathcal{R}, w', w'' \in \mathcal{V} \cup \mathcal{R}\}$. Each vertex has a representative, initially its unique variable. The representative of $V(w)$ is noted $\bar{w}$. We define the set of equations $R = \{w_1 = w_2 | w_1 \in \mathcal{V},\ w_2 \in \mathcal{V} \cup \mathcal{R},\ w_1 = val(w_2) \in \mathcal{E}\}$ (remind our convention about $\mathcal{E}$ above). Each equation e in R has a level l_e, initially set to 0. The algorithm iteratively computes the graph $U(\mathcal{E})$, an iteration of the while loop selects an equation and merges the corresponding vertices and edges. By taking care of the precedence, the algorithm successively gathers each equation in $\mathcal{E}$ into the partial graph $G = (V, E)$. The proofs of termination and correctness of this algorithm can be found in [5]. We define $\mathcal{V}_v = \mathcal{V} \cap v$ (resp. $\mathcal{R}_v$). A vertex v is $\mathcal{V}$-free iff $\mathcal{V}_v = \emptyset$. A path is a pair $p = (v_0, O)$, v_0 a vertex and O an occurrence $b_1 \cdots b_n$ such that if $v_i = V(w)$ and $b_{i+1} = k$, $0 \leq i < n$, then $suc(\bar{w})$ is defined and $v_{i+1} = V(w_k)$ where

$suc(\bar{w}) = f(w_0, w_1)$. The source (resp. target) of p is the vertex v_0 (resp. v_n). The length $|p|$ of the path is equal to n, its sets of variables are the unions $\mathcal{V}_p \doteq \bigcup_{v_i} \mathcal{V}_{v_i}$, $\mathcal{R}_p = \bigcup_{v_i} \mathcal{R}_{v_i}$. Although unification graphs are multigraphs, we shall unambiguously refer to a path by its sequence of vertices $v_0, \ldots, v_n$. A (fundamental) cycle c is a path such that $v_0 = v_n$ and $v_i \neq v_j$, $0 \leq i < j < n$. For each w in $\mathcal{R}_c$, we define the leaf occurrence $O(w, c)$ of $val(w)$ along c as $b'_1 \cdots b'_k$ where, if $w \in v_i$, $b'_j = b_{i+j}$ and the occurrence $b'_1 \cdots b'_j$ is an occurrence of $val(w)$. We also use the notation $O(M, c)$ if $M = val(w)$.

For each cycle c in $U(\mathcal{E})$, $\mathcal{V}_c$ is non-empty: the cycle c contains at least one vertex with a variable w. If w is in $\mathcal{V}$ we are done. Otherwise the term $val(w)/O(w, c)$ is a variable from $\mathcal{V}$ and belongs to the cycle. This merely restates the fact that a cycle corresponds to a positive occur-check in unification. A vertex v is a predecessor (resp. ancestor) of a vertex v' if there is an edge (resp. path) from v to v', resp. successor (resp. descendant). We also use the notation v/O meaning the target of the path (v, O) if well-defined. Let $e = (b, \bar{w}, \bar{w}')$ be an edge in $U(\mathcal{E})$. We say that e is incident to the vertex $V(w')$.

By an iteration we mean the execution of the body of the while-loop. When we refer to the values of some variable, we understand that this value is defined by the first occurrence of the variable encountered in this execution. Variables will be superscripted by iterations. Especially, the graph at i^{th} iteration is defined by $G^i = (V^i, E^i)$ and G^0 is the term dag representation $G(\mathcal{E})$ of $\mathcal{E}$. The equations m_1 (m_2) possibly created will be refered to as the left (right) equation created at iteration i. Notice that the representant $\bar{w}$ of w may change according to the iterations. However, due to the context, the notation is non-ambiguous. Let N denote the number of iterations. The level l_i of the i^{th} iteration is the level of the equation selected by this iteration, with $l_N = 0$. To an iteration $i < N$ we associate the first iteration $j > i$ such that $l_j \leq l_i$, this iteration is noted i^+. Especially, a 0 level iteration i selects an equation from $\mathcal{E}$, and i^+ is the first iteration following i that selects another equation from $\mathcal{E}$, or the N^{th} iteration.

We need some technical results relating the local structure of the graph $U(\mathcal{E})$ to the occurrences of variables in the equations $\mathcal{E}$. Let us start with some immediate observations.

- If $v \in V^i$ possesses two successors, a left one v' and a right one v'', then there exists $w \in \mathcal{R}_v$, $w' \in v'$, $w'' \in v''$ such that $suc(w) = f(w', w'')$;
- If $suc(w) = f(w_1, w_2)$, $suc(w') = f(w_3, w_4)$ and $V^N(w) = V^N(w')$, there are two equations m, m' in E_0 such that w (resp. w') occurs in the right-hand side of m (resp. m'). Let i be the first level 0 iteration such that m, m' do not belong to E_i, then $V^i(w) = V^i(w')$, $V^i(w_1) = V^i(w_3)$ and $V^i(w_2) = V^i(w_4)$.
- Let v be a vertex of G^i, then $\mathcal{R}_v \cup \mathcal{V}_v \neq \emptyset$. Let v' be a vertex of G^j, $j > i$, such that $v \subseteq v'$, the inclusion is proper iff v has been merged with some vertex between iterations i and j.
- Let v be a vertex of G^i such that $w_1 \neq w_2$ are in v. There exists an iteration $j < i$

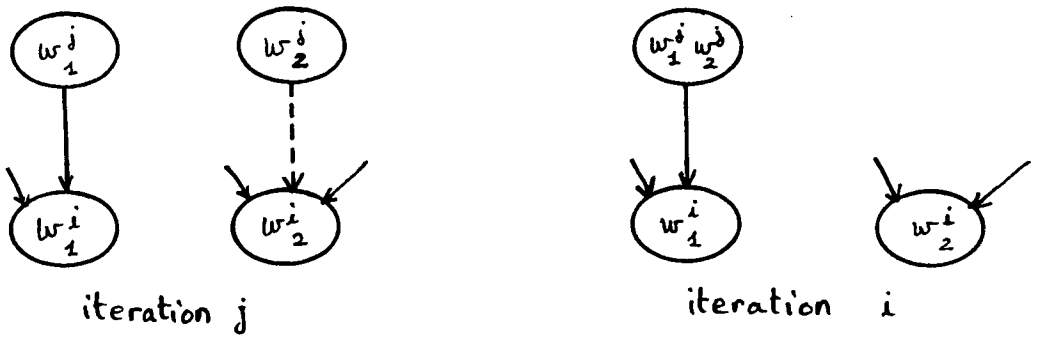

Figure 3: Creation and Selection of an equation.

that selects an equation $w = w'$ with $V^j(w) = V^j(w_1)$ and $V^j(w') = V^j(w_2)$ and $V^j(w_1) \neq V^j(w_2)$. Assume that $suc(w_1) = f(w_3, w_4)$ and $suc(w_2) = f(w_5, w_6)$, then either (i) $V^i(w_3) = V^i(w_5)$ or (ii) there exists $w'_3 \in V(w_3)$ and $w'_5 \in V(w_5)$ such that $w'_3 = w'_5$ belongs to E_i, with a positive level (resp. for w_4 and w_6).

- If the equation $m : w_1 = w_2$ is created at iteration j and selected at iteration i, then any equation created at iteration k such that $j < k < i$ is selected at iteration l such that $k < l < i$. Conversely, an equation selected at iteration l such that $j < l < i$ is created at iteration k such that $j < k < l$.

Definition 2.1 *Let $G = (V, E)$ be a graph and v be in V. The graph $G \downarrow v$ below v defined by (V_v, E_v) where V_v is the set of vertices v' such that there exists a path with source v and target v', and E_v is the set of edges whose both source and target are in V_v. The graph $G \uparrow v = (V^v, E^v)$ above v is defined by $V^v = (V - V_v) \cup \{v\}$ and $E^v = \{(i, w, w') \in E \mid V(w), V(w') \in V^v\}$.*

The two edges $e_i = (b_i, w_i, w'_i) \in E^{j_i}$, $i = 1, 2$, are congruent, noted $e_1 \simeq e_2$, iff $b_1 = b_2$ and the source and target of one edge are respectively included in the source and target of the other edge.

Every edge in E^{i+1} possesses a congruent edge in E^i. The following observation will be useful throughout the paper. Between a level 0 iteration i and the level 0 iteration i^+ the algorithm at least merges an "equation" from $\mathcal{E}$ to G^i (the leafs of the term dag associated to the right-hand side of the equation can be the root of an already computed subgraph). The graph $G \downarrow V^i(w_2)$ "starts" at its root $V^i(w_2)$ with the term dag $val(w_2)$. The non-leaf vertices of the tree are $\mathcal{R}$-singletons, the internal ones possess a unique predecessor, and, if $w_2 \in \mathcal{R}$, the vertex $V^i(w_2)$ is a root (remind our convention on $\mathcal{E}$ at the beginning of this subsection). We first establish two easy lemmas relating vertices of an equation that has been created at some iteration and the vertices of this equation at the iteration that selects it.

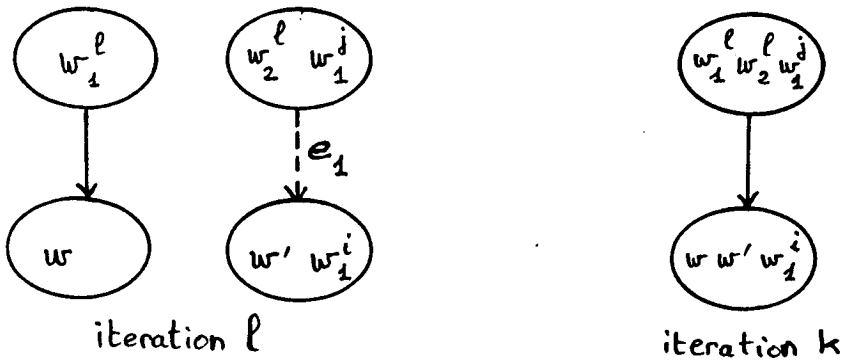

Figure 4: Deletion of edge e_1 in Lemma 2.1.

Lemma 2.1 *Assume that the equation $m : w_1^i = w_2^i$ is created at iteration j and selected at iteration i, then $V^i(w_1^i)$ possesses $V^i(w_1^j) = V^i(w_2^j)$ as predecessor.*

Proof. Cf. Fig. 4. If $i = j+1$ or if the edge $e_1 = (b, \bar{w}_1^j, \bar{w}_1^i)$ is not deleted between iterations j and i, this is immediate. Otherwise, if the edge e_1 is deleted at iteration l, $j < l < i$, there exist w that occurs in $suc(\bar{w}_1^l)$, w' in $suc(\bar{w}_2^l)$, such that $m_l : w = w'$ is created at iteration l, $V^l(w_1^i) = V^l(w')$ and $V^l(w_1^j) = V^l(w_2^l)$. We have $V^{l+1}(w_1^l) = V^{l+1}(w_1^j) = V^{l+1}(w_2^l)$ and $e_2 = (b, \bar{w}_1^j, \bar{w}) \in E^{l+1}$. The equation m_l is selected strictly before iteration i. Also, there exists an iteration k, $l < k \leq i$, such that the edge $e_3 = (b, \bar{w}_1^j, \bar{w}_1^i) \in E^k$ is incident to $V^k(w_1^i) = V^k(w') = V^k(w)$. That is $e_1 \simeq e_2 \simeq e_3$. In turn, if this last edge is not deleted we get the result. Otherwise the same reasoning applies and some edge $e_n = (b, \bar{w}, \bar{w}_1^i)$, congruent to e_1, is incident to $V^i(w_1^i)$. □

Corollary 2.2 *Assume that the equation $m : w_1^i = w_2^i$ is created at iteration j and selected at iteration i, then the vertex $V^{i+1}(w_1^i) = V^{i+1}(w_2^i)$ possesses as predecessor the vertex $V^{i+1}(w_1^j) = V^{i+1}(w_2^j)$.*

Lemma 2.3 *Assume that the equation $m : w_1^i = w_2^i$ is created at iteration j, selected at iteration i, and that $V^j(w_2^i)$ possesses a unique incident edge, then $V^j(w_2^i) = V^i(w_2^i)$.*

Proof. The edge $(b, \bar{w}_2^j, \bar{w}_2^i)$ is deleted at iteration j, and is by hypothesis the only edge incident to $V^j(w_2^i)$. If $i = j + 1$, the result is true. Otherwise, at iteration $j + 1$, the other equation created at iteration j is selected. Assume that $V^j(w_2^i) \neq V^i(w_2^i)$ and let k, $j < k \leq i$, be the first iteration such that $V^j(w_2^i) \neq V^k(w_2^i)$. The equation $m : w = w'$ selected at iteration $k - 1$ is such that $V^{k-1}(w)$ or $V^{k-1}(w')$ is $V^{k-1}(w_2^i) = V^j(w_2^i)$. But equation m has been created at iteration l such that $j < l < k - 1$. Also, at iteration l, $V^l(w_2^i)$ is not a root, while $V^j(w_2^i)$ is. This is possible only if $V^l(w_2^i) \neq V^j(w_2^i)$, which contradicts the minimality of k. □

Corollary 2.4 *Assume that the equation $m : w_1^i = w_2^i$ is created at iteration j and selected at iteration i, then every edge incident to $V^j(w_l^i)$ is congruent to an edge incident*

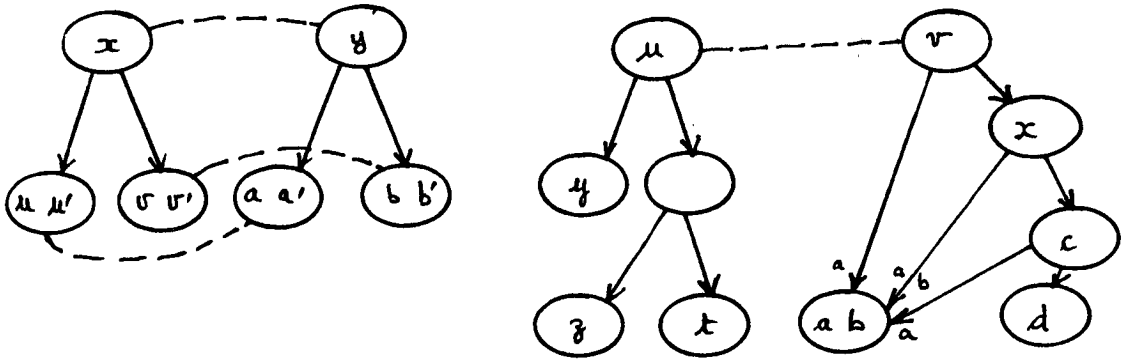

Figure 5: Graphs of sets $\mathcal{E}_3$ and $\mathcal{E}_4$.

to $V^i(w_l^i)$, $l = 1,2$, *with the exception of the edge* $(b, V^j(w_2^j), V^j(w_2^i))$ *that creates the equation* m.

Lemma 2.5 *Let* v *be a root of* V^N, *then* $\forall x, y \in \mathcal{V}_v$, $x =_s y$. *Let* v *be a non-root vertex of* V^N, *then* $\forall x \in \mathcal{V}_v\ \exists y \in \mathcal{V}_v$, $x =_s y$ *and* y *occurs by some edge incident to* v.

Proof. Let i be the first iteration such that $V^i(x) = V^i(y)$. The equation selected at iteration i is of the form $x' = y'$, $x', y' \in \mathcal{V}$ and belongs to $\mathcal{E}$. If $x \not\equiv x'$ or $y \not\equiv y'$, say $x \not\equiv x'$, we consider the first iteration j, $j < i$, such that $V^j(x) = V^j(y)$ and so on.

Assume that $V^0(x)$ is a root. Consider the first iteration i such that $V^{i+1}(x)$ is non-root. The equation selected at iteration i belongs to $\mathcal{E}$. As the vertex merged with $V^i(x)$ is non-root, the equation is strict, say $x = x'$. In turn, if $V^0(x')$ is a root, then we consider the first iteration j, $j < i$, such that $V^j(x')$ is non-root and so on. □

Lemma 2.5 is false of arbitrary iterations of the unification algorithm. However it is easily seen to be true at level 0 iterations. This shows that some care is needed in these technicalities. Consider the two sets:

$$\mathcal{E}_3 \left\{ \begin{array}{lcl} 1: & x & = f(u,v) \\ 2: & x & = f(u',v') \\ 3: & y & = f(a,b) \\ 4: & y & = f(a',b') \\ 5: & x & = y \end{array} \right. \qquad \mathcal{E}_4 \left\{ \begin{array}{lcl} 1: & u & = f(y, f(z,t)) \\ 2: & v & = f(a,x) \\ 3: & x & = f(a,c) \\ 4: & x & = f(b, f(b,d)) \\ 5: & u & = v \end{array} \right.$$

If the algorithm selects the equations in this order, then both part of Lemma 2.5 are false at some iteration. The set $\mathcal{E}_3$ invalidates the first proposition at some iteration i by $a \not\equiv_s a'$ and $V^i(a) = V^i(a)$ is a root; while for $\mathcal{E}_4$, if the equation $x = f(z,t)$ is selected before $y = a$, the second proposition is false for a and the non-root vertex $V^j(a)$ at some iteration j.

Let $e = (b, w, w')$ be an edge incident to a vertex v. We say that $x \in \mathcal{V}_v$ occurs by e iff there exists $w'' \in V(w) \cap \mathcal{R}$ such that $suc(w'') = f(x, w''')$ and $b = 0$, or $suc(w'') =$

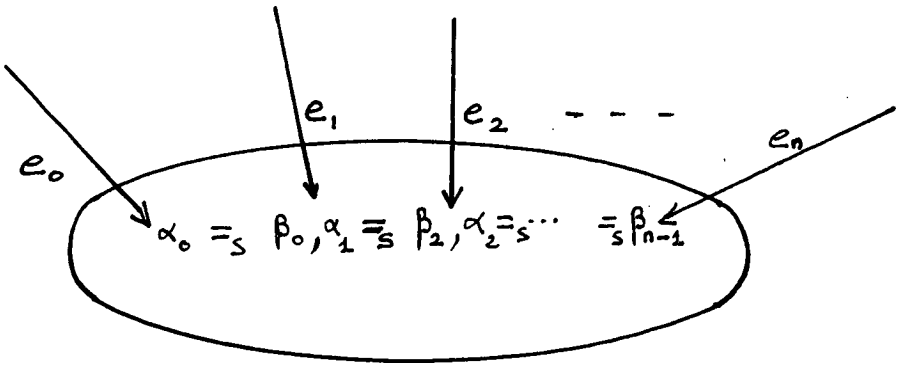

Figure 6: A Chain between two co-incident edges.

$f(w''', x)$ and $b = 1$. Assume that x occurs by e in $\mathcal{V}_v$. An occurrence of x in some non-strict right-hand side $m : y = C[x]$ is said to be associated to e iff the edge e is the last edge of the path $p = (V(y), O_C)$. A vertex is shared iff it is the target of two distinct edges. Notice that if v is shared, x can occur by distinct edges. The next Proposition introduces chains of variables, these are sequences of variables that possess multiple occurrences and belong to the same vertex. A weak consequence of the second part of this Proposition is the unique incident edge property for $\mathcal{V}$-free vertices. The chains are described in Fig. 6, they play a central rôle in completeness and in §5.

Proposition 2.6 *Let E be a set of equations, and i be some iteration, then:*

1. *For all edges $e = (b, w, w')$ in E^i such that $V^i(w') \cap \mathcal{V} \neq \emptyset$, there exists a variable x in this set that occurs by e.*
2. *For all vertices v in V^i and for all pairs (e, e') of distinct edges in E^i incident to v, there exists a sequence of pairs (x_j, y_j) of variables in $\mathcal{V}_v$, $j = 0, \ldots, n-1$, and a sequence of edges (e_k) incident to v, $k = 0, \ldots, n$, $e_0 = e$, $e_n = e'$, such that $x_j =_s y_j$, x_j occurs by e_j and y_j occurs by e_{j+1}.*

Proof. The two propositions are simultaneously proved by induction on the iterations. Their truth at iteration 0 is immediate.

Assume the two propositions true at iterations $j < i + 1$. We first prove 1 at iteration $i + 1$. Let the edge e in E^{i+1} be such that $V^{i+1}(w')$ is not $\mathcal{V}$-free. The two non-trivial cases are:

i. $e \simeq (b, w, \bar{w}_1^i) \in E^i$, $V^i(w_1^i)$ is $\mathcal{V}$-free, $V^i(w_2^i)$ is not $\mathcal{V}$-free.

ii. $e \simeq (b, w, \bar{w}_2^i) \in E^i$, $V^i(w_2^i)$ is $\mathcal{V}$-free, $V^i(w_1^i)$ is not $\mathcal{V}$-free.

Let j, if it exists, be the iteration that creates the equation $m : w_1^i = w_2^i$ selected at iteration i.

In case **i**, the equation m cannot be of level 0, for this implies $w_1^i \in \mathcal{V}$. Also $j > 0$. We claim that $V^j(w_2^i) \cap \mathcal{V} \neq \emptyset$. Otherwise, by 2 true at j by induction hypothesis, this

vertex possesses a unique incident edge, and by Lemma 2.3, we have $V^j(w_2^i) = V^i(w_2^i)$ which is $\mathcal{V}$-free, contradiction. Hence we may apply 1 to the edge $e' = (b, \bar{w}_2^j, \bar{w}_2^i) \in E^j$ with $V^j(w_2^i) \cap \mathcal{V} \neq \emptyset$. There exists x in $V^j(w_2^i)$, w'' in $V^j(w_2^j) \cap \mathcal{R}$ such that x occurs in $suc(w'')$, according to e'. But $V^j(w_1^i)$, being $\mathcal{V}$-free, has a unique incident edge. Now, by Corollary 2.2 the edges e and e' are congruent. This proves 1 for the edge e at iteration $i+1$.

We prove that case **ii** is self-contradictory. The equation m cannot be of level 0. Otherwise, it would be strict as $V^i(w_2^i)$ is not a root. But this strictness implies that this vertex is not $\mathcal{V}$-free. Thus equation m has been created at some iteration $j > 0$. The vertex $V^j(w_2^i)$ is $\mathcal{V}$-free. Hence by 2 and Lemma 2.3, $V^j(w_2^i) = V^i(w_2^i)$. Hence no iteration between j and i modifies the vertex $V^j(w_2^i)$. Consequently the edge $(b, w, \bar{w}_2^i) \in E^i$ also belongs to E^j, perhaps with w replaced by some w'' such that $V^j(w'') \subseteq V^i(w)$. Thus, $V^j(w_2^i)$ possesses two distinct incident edges. By induction hypothesis 2, this vertex is not $\mathcal{V}$-free, which is again a contradiction.

We prove 2 at iteration $i+1$. Let $v \in V^{i+1}$ and the two distinct incident edges e_1 and e_2 be in E^{i+1}. We consider the single non-trivial case:

$$\begin{cases} e_1' = (b_1, w_1, \bar{w}_1^i) \in E^i, & e_1' \simeq e_1 \\ e_2' = (b_2, w_2, \bar{w}_2^i) \in E^i. & e_2' \simeq e_2 \end{cases}$$

If the equation $m : w_1^i = w_2^i$ is of level 0, it is strict as $V^i(w_2^i)$ is not a root. We apply 1 to the edges e_1' and e_2', this gives us by induction hypothesis x and y in $\mathcal{V}$ that occur by e_1' and e_2' respectively. By Lemma 2.5, true at level 0 iterations, applied to the two non $\mathcal{V}$-free vertices $V^i(w_1^i)$, $V^i(w_2^i)$, and to the $\mathcal{V}$-variables w_1^i, w_2^i, we get $x_1 =_s w_1^i$ and $y_1 =_s w_2^i$, where x_1 occurs by some edge e_1'' and y_1 by some edge e_2''. Hence $x_1 =_s y_1$ By induction hypothesis applied to the pairs of edges (e_1', e_1'') and (e_2'', e_2'), we have two sequences of edges and pairs of variables according to 2. They can be appended by $x_1 =_s y_1$ and give the result for iteration $i+1$.

Otherwise the equation m is not of level 0 and is created at iteration j. We know that all edges incident to $V^j(w_l^i)$ are also incident to $V^i(w_l^i)$, $l = 1, 2$, by Corollary 2.4 with the usual proviso.

If both e_1' and e_2' are such edges, we apply the induction hypothesis at iteration j to the two vertices $V^j(w_1^i)$ and $V^j(w_2^i)$, with respectively the pairs of edges (e_1', e_3), $e_3 = (b, \bar{w}_1^j, \bar{w}_1^i)$, and (e_4, e_2'), $e_4 = (b, \bar{w}_2^j, \bar{w}_2^i)$. The sequences thus obtained give the sequence for iteration $i+1$, by Corollary 2.2 and the congruences $e_3 \simeq (b, \bar{w}_1^j, \bar{w}_1^i)$ and $e_4 \simeq (b, \bar{w}_1^j, \bar{w}_1^i)$, true at $i+1$.

Otherwise, by Lemma 2.3, the vertex $V^j(w_2^i)$ possesses at least two incident edges $e_3 = (b, \bar{w}_2^j, \bar{w}_2^i)$ and e_4. Applying 2 to the pair of edges (e_3, e_4) at iteration j gives us a sequence s_2. By Corollary 2.4, the edge e_4 is congruent to and edge e_4' incident to $V^i(w_2^i)$. Applying 2 at iteration i to the pair (e_4', e_2') gives a sequence s_2. Let k, $j \leq k \leq i$, be the first iteration such that e_1' is incident to $V^k(w_1^i)$. We have two cases:

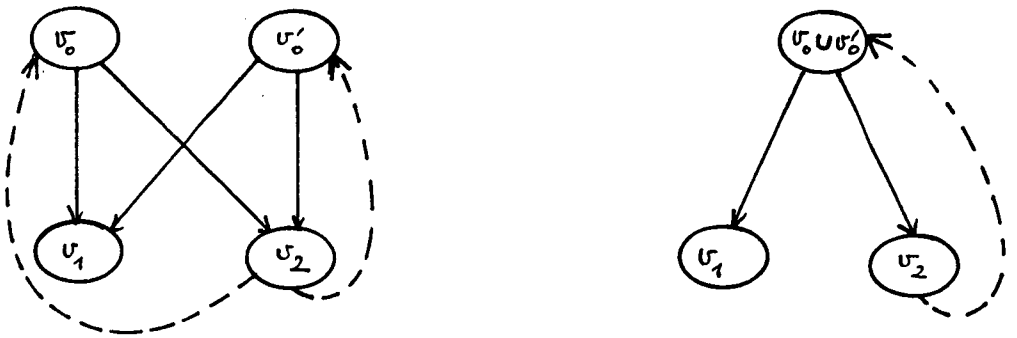

Figure 7: Merging two vertices with the same sons.

- e_1' is incident to $V^{k-1}(w_2^{k-1}) \neq V^{k-1}(w_1^i) = V^{k-1}(w_1^{k-1})$. By Lemma 2.1, $V^{k-1}(w_1^{k-1})$ possesses at least one incident edge e_5. We apply 2 at iteration k to the pair (e_1', e_5), this gives us a sequence s_3. We iterate the construction with e_5, either incident to $V^j(w_1^i)$ or not. This gives a list of sequences s_j, $j = 3, \ldots, n$, with s_n the sequence for a pair of edges (d, d') such that $d' = (b, \bar{w}_1^j, \bar{w}_1^i) \in E^j$ and $d \simeq e_1' \simeq e_1$. The concatenation of the sequences $s_3 \cdots s_n s_1 s_2$ is the required sequence.

- e_1' is incident to $V^{k-1}(w_1^{k-1}) \neq V^{k-1}(w_2^{k-1}) = V^{k-1}(w_1^i)$. We have two subcases. Firstly, $e_1' \simeq (b, \bar{w}_1^j, \bar{w}_1^i)$, then $s_1 s_2$ is the required sequence. Secondly, we consider the first iteration l, $k \leq l < i$, such that the edge $e_6 = (b, \bar{w}_1^j, \bar{w}_1^i)$ is incident to $V^l(w_1^i)$. This iteration is well-defined by Lemma 2.1. Applying 2 to the pair (e_1', e_6) gives a sequence s_3. The required sequence is $s_3 s_1 s_2$. □

2.4. Completeness. Turning to the cycles in a unification graph, the above results shed some light on their local structure. However, we do not yet know how the cycle is constructed from the equations in $\mathcal{E}$. Considering the graph $U(\mathcal{E})$ as our model, the answer to this question is provided by a deduction in the equational logic LE. Also we establish (constructive) completeness results. Before, we precise the relationship between the two graphs $U(\mathcal{E})$ and $W(\mathcal{E})$ with respect to unifiability. Both Lemma 2.5 and Proposition 2.6 fail for the graph $W(\mathcal{E})$ (cf. the set $\mathcal{E} = \{x = f(z,t),\ y = f(z,t),\ u = f(x,w),\ v = f(y,w)\}$). However, only this last graph has logical meaning, as is clear from the Completeness Theorem below. Also, in the study of obstruction to unifiability, we have to justify the restriction to $U(\mathcal{E})$.

Lemma 2.7 *The graph $W(\mathcal{E})$ is acyclic iff the graph $U(\mathcal{E})$ is acyclic.*

Proof. The Lemma is immediate from the construction of $W(\mathcal{E})$ from $U(\mathcal{E})$. Consider the merging of two vertices when computing $W(\mathcal{E})$ from $U(\mathcal{E})$. This operation is local and Fig. 7 shows that if the resulting graph is cyclic, so is the initial graph.□
For completeness, in a first step, under the hypotheses $\mathcal{E}$, an equational deduction $\mathcal{D} \vdash_{LE_0}$

$M = N$ will be associated to each pair of distinct subterms M and N of $\mathcal{E}$ in a vertex of $U(\mathcal{E})$. Afterwards, to any path p will be associated a set of deductions $\mathcal{D} \vdash_{LE_0} x = C[y]$, $C[\_]$ a non-trivial context, $x \in \mathcal{V}_p$, $O_C = O(C[y], p)$. The set of hypotheses of $\mathcal{D}$ is noted $\mathcal{A}(\mathcal{D})$. The path completeness for the restricted system LE_0 will be proved via the following system, obviously equivalent to LE_0 (they define the same class of theorems):

$$(s)\ \frac{M = N}{N = M} \qquad (t)\ \frac{M = x \quad x = N}{M = N}$$

$$(sl)\ \frac{f(M_1, N_1) = f(M_2, N_2)}{M_1 = M_2} \qquad (sr)\ \frac{f(M_1, N_1) = f(M_2, N_2)}{N_1 = N_2}$$

$$(su)\ \frac{x = M \quad y = C[x]}{y = C[M]}$$

The following derived rules will be useful:

$$(dl)\ \frac{x = f(M_1, N_1) \quad x = f(M_2, N_2)}{M_1 = M_2} \qquad (dr)\ \frac{x = f(M_1, N_1) \quad x = f(M_2, N_2)}{N_1 = N_2}$$

The use of Proposition 2.6 and Lemma 2.5, although proved for sets of equations in simplified form $x = M$, is valid for general equations by the presence of the rules (sl) and (sr). We first establish the soundess and completeness of this system for equations holding between subterms in unification equivalence classes. With the notations of §2.3:

Lemma 2.8 *Let $w_1 \neq w_2$ be two variables in $\mathcal{V} \cup \mathcal{R}$, then $\mathcal{D} \vdash_{LE_0} val(w_1) = val(w_2)$, where $\mathcal{D}$ is (su)-free, iff $V(w_1) = V(w_2)$ in $U(\mathcal{A}(\mathcal{D}))$.*

Proof. The proof of adequation is straighforward by structural induction on deductions. The proof of completeness is by induction on the cardinality of $\mathcal{E} = \mathcal{A}(\mathcal{D})$. With a single equation $m : x = M$, the only non-singleton vertex is $V(x)$, the deduction reduces to an hypothesis. Assume the lemma true for $\mathcal{E}$, we add an equation $m : x = M$. If x does not occur in $U(\mathcal{E})$, there is nothing to prove. Otherwise, in $V(x)$ we have $\mathcal{D} \vdash_{LE_0} val(w_1) = val(w_2)$ for all pairs of distinct variables w_1 and w_2 by induction hypothesis. Especially, $\mathcal{D} \vdash_{LE_0} val(w_1) = x$, which gives the deduction $\mathcal{D}'$:

$$(t)\ \frac{\begin{array}{c}\mathcal{D} \\ val(w_1) = x \quad x = M\end{array}}{val(w_1) = M}$$

and $\mathcal{D}'' \vdash_{LE_0} M = val(w_1)$ with an instance of the symmetry rule. If $M \in \mathcal{V}$ or if the vertex $V(x)$ is not predecessor, there is nothing more to prove.

Otherwise, $M = f(M_1, M_2)$ and $V(x)$ has two successors. We construct new deductions, say for the left successor v and the term M_1.

If $\mathcal{V}_v \neq \emptyset$, by Proposition 2.6 there exists $w \in \mathcal{R} \cap \mathcal{V}(x)$ and $y \in \mathcal{V}_v$ such that $suc(w) = f(y, w')$ for some w'. By induction hypothesis there exists a deduction $\mathcal{D} \vdash_{LE_0} x = f(y, val(w'))$. We have a deduction $\mathcal{D}'$:

$$(t)\ \frac{\begin{array}{c}\mathcal{D} \\ x = f(y, val(w')) \quad x = f(M_1, M_2)\end{array}}{y = M_1}$$

Now, for all w'' in v, we have deductions for the equations $val(w'') = M_1$ and $M_1 = val(w'')$, with $\mathcal{D}'$, transitivity and symmetry.

Otherwise, $\mathcal{V}_v$ is empty. By Proposition 2.6, the vertex v possesses a unique incident edge. Consequently, for all w_1 in v, w_1 is in $\mathcal{R}$ and there exists w_0 in $\mathcal{R} \cap \mathcal{V}^{\mathcal{E}}(x)$ such that $suc(w_0) = f(w_1, w_2)$ for some w_2. We apply the rule (sl) as above.

In turn, if v is not predecessor or if $M_1 \in \mathcal{V}$, there is nothing more to prove. Otherwise the same proof is carried on. □

Next, we are interested in path deductions: for $p = (V(x), O)$ such that both its source and target are not $\mathcal{V}$-free, these are deductions of the form $\mathcal{D} \vdash_{LE_0} x = C[y]$ with $O_C = O$. The substitution rule allows us to combine deductions given by Lemma 2.8.

Lemma 2.9 *Let $p = v_0, \ldots, v_n$ be a path such that $\mathcal{V}_{v_i} = \emptyset$ for $0 < i < n = |p|$, $\mathcal{V}_{v_0} \neq \emptyset$ and $\mathcal{V}_{v_n} \neq \emptyset$. For all $x \in \mathcal{V}_{v_0}$ and for all $y \in \mathcal{V}_{v_n}$, there exist a variable $w \in \mathcal{R}_{v_0}$, a deduction $\mathcal{D} \vdash_{LE_0} x = val(w)$ with $val(w) = C[y]$, $O_C = O(w, p)$ and $|O_C| = n$.*

Proof. By Proposition 2.6 there exists w_1 in $\mathcal{R}_{v_{n-1}}$ and y in $\mathcal{V}_{v_n}$ such that $suc(w_1) = f(y, w')$ or $suc(w_1) = f(w', y)$. By the unique incident edge property for $\mathcal{V}$-free vertices, there exists w in v_0 such that $val(w) = C[y]$, $O_C = O(p, w)$. The conclusion follows from Lemma 2.8. □

The next Proposition is the origin of the normal form result of §3. Its proof is constructive and suggests the existence of a normal form result.

Proposition 2.10 *Let $p = (v_0, O) = v_0, \ldots, v_n$ be a path such that both $\mathcal{V}_{v_0}$ and $\mathcal{V}_{v_n}$ are non-empty, $n = |p|$, then for all $x \in \mathcal{V}_{v_0}$ and for all $y \in \mathcal{V}_{v_n}$, there exists a deduction $\mathcal{D} \vdash_{LE_0} x = C[y]$ with $O_C = O$. Conversely, for all deductions $\mathcal{D} \vdash_{LE_0} x = C[y]$, there exists a path from $V(x)$ to $V(y)$ in $U(\mathcal{A}(\mathcal{D}))$.*

Proof. The deduction is built bottom up by recursive applications of Proposition 2.6, Lemmas 2.5 and 2.8. Let e be the last edge of the path p, incident to v_n. As $\mathcal{V}_{v_n} \neq \emptyset$, there exists a variable y' that occurs in v_n by e. Let $\mathcal{D}^1 \vdash_{LE_0} y' = y$ and $z = C[y']$ be some (non-strict) equation associated to the occurrence of y' by e. We have three cases according to the relative positions of the paths p and $p' = (V(z), O_C)$ (cf. Fig. 8):

1. p' is a proper suffix path of p. We iterate the construction with z and the proper prefix path $v_0, \ldots, V(z)$ of p.

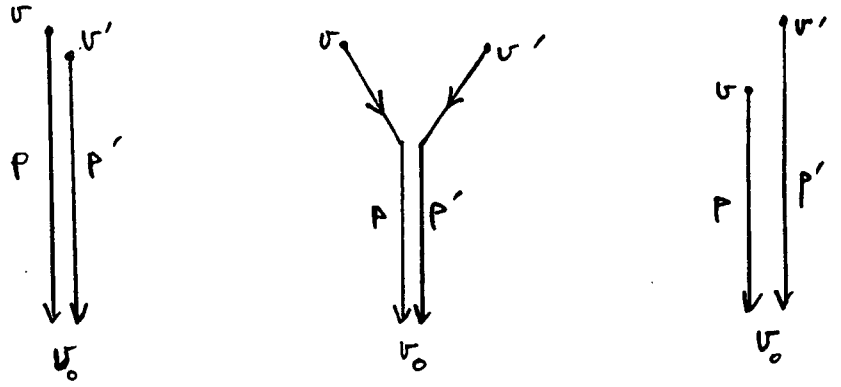

Figure 8: Relative positions of two co-incident paths.

2. p is a suffix path of p'. There exists a deduction $D^2 \vdash_{LE_0} x = C'[y']$ where $x = C'[y']$ is a subterm occurrence of $C[y']$.

3. The paths diverge at vertex v_k, $0 < k < n$. This vertex is shared and by Proposition 2.6 there exists some variable z' that occurs in v_k by the edge of p incident to v_k. We have a deduction $D^3 \vdash_{LE_0} z' = C''[y']$ for $C''[y']$ a subterm occurrence of $C[y']$. We iterate the construction with z' and the proper prefix path $v_0, \ldots, v_k$ of p.

The second case is the halting configuration which is eventually reached. Collecting the subdeductions along the recursive calls, the inference tree has the following form:

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{D}_2}{x_2 = C_2[x_3]} \quad (su)\ \dfrac{\dfrac{\mathcal{D}_1}{x_1 = C_1[x_2]} \quad \dfrac{\mathcal{D}_0}{x = C_0[x_1]}}{x = C_0C_1[x_2]}}{x = C_0C_1C_2[x_3]} \quad \cdots}{x = C_0 \cdots C_{m-1}[x_{m-1}])} \quad (su)\ \dfrac{\dfrac{\mathcal{D}_{m-1}}{x_{m-1} = C_{m-1}[y]} \quad \vdots}{x = C_0 \cdots C_m[y]}$$

where each deduction $\mathcal{D}_i$, $i = 0, \ldots, m-1$, is (su)-free and auxiliary (its conclusion is the non-principal premiss of the introduction rule (su)). □

The reader may wish to find some equational deductions for the example $\mathcal{E}_M$ of the introduction. This corollary allows us to restrict ourselves to deductions of the form specified in its proof. This (informal) normal form result for LE_0 will have its counterpart for the system LE.

Corollary 2.11 *For each cycle c in $U(\mathcal{E})$ and each variable x in $\mathcal{V}_c$, there exists a deduction of a cyclic equation $\mathcal{D} \vdash^{\mathcal{E}}_{LE_0} x = C[x]$ with $|O_C| = |c|$, $O_C = O(c, C[x])$, and conversely.*

We have established the existence of a deduction $\mathcal{D}_c \vdash_{LE_0} x = C[x]$ associated to a cycle c. For practical computations, we record the usefulness of a system LE_1 for the type inference problem (cf. §6).

Proposition 2.12 *The following inference system LE_1 is sound and complete for path deductions in $U(\mathcal{E})$, provided that equations in $\mathcal{E}$ are in simplified form $x = M$:*

$$(s)\ \frac{x = y}{y = x} \qquad (t)\ \frac{x = y \quad y = M}{x = M}$$

$$(d)\ \frac{x = C_1[y] \quad x = C_2[M] \quad C_1[\_] \sim C_2[\_]}{y = M} \qquad (su)\ \frac{x = M \quad y = C[x]}{y = C[M]}$$

Proof. Soundness is immediate. Completeness is an exercice on rewriting systems. Any deduction under the first inference system, in the form given by Proposition 2.10, can be transformed into a deduction under the above inference system according to a rewriting process. We omit reductions involving the derived rules (dl) and (dr). First, instances of the symmetry rule can be lifted up to hypotheses. Next, instances of the rule (t) such that the leftmost term of the premisses is of positive size cannot be the last inference of a deduction for an equation of the form $x = M$. Further, it can only be followed by the rules (t), (sl) or (sr). This allows us to define a reduction on deductions that contain such instances of (t). We give three rules, the others are deduced by symmetry. In each rule M_1 is of positive size and we have $M_1 \equiv f(M_1^1, M_1^2)$ as well as $M_2 \equiv f(M_2^1, M_2^2)$ in the last two rules.

$$(t)\ \frac{(t)\ \dfrac{M_1 = x \quad x = y}{M_1 = y} \quad y = M_2}{M_1 = M_2} \quad \Longrightarrow \quad (t)\ \frac{M_1 = x \quad (t)\ \dfrac{x = y \quad y = M_2}{x = M_2}}{M_1 = M_2}$$

$$(sl)\ \frac{(t)\ \dfrac{(s)\ \dfrac{x = M_1}{M_1 = x} \quad x = M_2}{M_1 = M_2}}{M_1^1 = M_2^1} \quad \Longrightarrow \quad (dl)\ \frac{x = M_1 \quad x = M_2}{M_1^1 = M_2^1}$$

$$(sl)\ \frac{(t)\ \dfrac{(sl)\ \dfrac{f(M_1, M_3) = f(x, M_4)}{M_1 = x} \quad x = M_2}{M_1 = M_2}}{M_1^1 = M_2^1} \quad \Longrightarrow \quad (dl)\ \frac{(sl)\ \dfrac{(s)\ \dfrac{f(M_1, M_3) = f(x, M_4)}{f(x, M_4 =) f(M_1, M_3)}}{x = M_1} \quad x = M_2}{M_1^1 = M_2^1}$$

In the last two reductions, the existence of an inference above the left premiss of (t) exists due to the form of the hypotheses. We used the derived rules. Redexes containing them are easily deduced from the above reductions. Confluence is easy to check (by critical pairs), and well-foundedness follows from the observation that the number of (t)-rules whose left premiss is of positive size strictly decreases. By inspecting normal forms, we see that: 1) no (s)-rule is applied to a non-strict hypotheses; 2) instances of (t) have been reduced to instances of (t) according to the second system; 3) any subdeduction involving only rules (dl), (dr), (sl) and/or (sr) has its conclusion of the form $x = t$ for $x \in \mathcal{V}$. Such a subdeduction starts necessarily by an instance of (dl) or (dr). Hence, such a deduction is an instance of the rule (d). This establishes the completeness of the second system. □
Completeness for LE now follows directly from path completeness of LE_0.

Theorem 2.13 $\vdash^{\mathcal{E}}_{LE} M = N$ *iff* $\models^{\mathcal{E}}_{W} M = N$.

Proof. As usual soundness follows from an easy induction on deductions. Completeness: if a term M is embeddable in $W(\mathcal{E})$, then it can be embedded in $U(\mathcal{E})$ as well. In the terminolgy of Fig. 8, this follows from the fact that if M is embedded in G', then it is embeddable in G. In turn, this is a consequence of the fact that $G(M)$ is a term dag, hence every internal vertex of $G(M)$ is unshared, hence has a unique incident edge. So that we have locally: if the embedding $\phi : G(M) \to G'$ possesses the edges e, e' in its range then at most one edge among those incident to the common source of e, e' in G' is (locally) in the range of ϕ. Consequently, the projection $p : G \to G'$ locally splits. This gives the desired embedding $\psi : G(M) \to G$ with $p \circ \psi = \phi$.

Also, given $\models^{\mathcal{E}}_{W} M = N$, we have at least two embeddings $\phi : G(M) \to U(\mathcal{E})$ and $\psi : G(N) \to U(\mathcal{E})$, possibly with distinct roots $V_U(M) \neq V_U(N)$. In the vertex $V_U(M)$, path completeness of LE_0 gives a deduction $\mathcal{D}_0 \vdash^{\mathcal{E}}_{LE_0} val(w'_1) = val(w_1)$, with the notations of §2.4. Each maximal occurrence O of $M \wedge M_0$, where $M_0 \equiv val(w'_1)$, such that $M/O \not\equiv M_0/O$, gives some deduction $\vdash^{\mathcal{E}}_{LE_0} val(w_2) = M_0/O$, which, associated to $\mathcal{D}_0$ by the left introduction of LE, gives $\mathcal{D}_1 \vdash M_1 = val(w_1)$ (notice the possibility that the proper term of this introduction is non-variable). And so on with $M \wedge M_1$ until we get $\mathcal{D} \vdash M = val(w_1)$. Similarly, we have $\mathcal{D}' \vdash val(w_2) = N$ for some $\mathcal{R}$-variable in $V_U(N)$'s root. If $V_U(M) = V_U(N)$, there is nothing more to prove. Otherwise, we consider a minimal sequence of vertices merging, which computes from $U(\mathcal{E})$ a quotient graph G with projection $p : U(\mathcal{E}) \to G$ such that the roots of $p(\phi(G(M)))$ and $p(\psi(G(N)))$ are identified. This sequence, being minimal, defines a term O with two distinct embeddings ϕ' and ψ' in $U(\mathcal{E})$, such that $\phi'(G(O))$ and $\phi(G(M))$, $\psi'(G(O))$ and $\psi(G(N))$, have equal roots. As above, we have $\mathcal{F} \vdash_{LE} val(w_3) = O$, with $V(w_3) = V(w_1)$, and $\mathcal{F}' \vdash_{LE} O = val(w_4)$, with $V(w_4) = V(w_2)$, By LE_0 path completeness, we have $\mathcal{H} \vdash val(w_1) = val(w_3)$ and $\mathcal{H}' \vdash val(w_4) = val(w_2)$. We conclude by collecting by transitivity the deductions $\mathcal{D}$, $\mathcal{D}'$, $\mathcal{F}$, $\mathcal{F}'$, $\mathcal{H}$ and $\mathcal{H}'$. □

Notice that the necessity, in the above proof, of the third term O, gives an informal explanation of the complexity of the normal forms in LE (cf. §3), with respect to those in LE_0. Also, the proof establishes the decidability of LE. This does not contradict the undecidability of equational logic. Rather, remember that everything, including substitutions and instances of reflexivity, is explicited in the hypotheses of deductions in LE. Here are easy counter-examples to other "completeness" attempts: with $x = f(u, v)$ and $y = f(u, v)$, we have $\vdash^{\mathcal{E}}_{LE} x = y$ but not $\models^{\mathcal{E}}_{U} x = y$; with $a = f(f(u, v), y)$ and $a = f(x, f(m, n))$, we have $\vdash^{\mathcal{E}}_{LE} a = f(x, y)$ and $\models^{\mathcal{E}}_{U} a = f(x, y)$, but not $\vdash^{\mathcal{E}}_{LE_0} a = f(x, y)$.

3 A Geometric Interpretation and Normalization

In this section, we work exclusively on the term dag $G(\mathcal{E})$ associated to a set of equations $\mathcal{E}$. In our framework, computation means getting some (topological) properties of the

space $W(\mathcal{E})$, by working on the data space $G(\mathcal{E})$, which is the *presentation* of $W(\mathcal{E})$. In other words, the previous section was concerned with the truth or satisfiability of equations, by analysis of the graph $W(\mathcal{E})$. The present and following sections are devoted to the study of the process by which the truth-value in $W(\mathcal{E})$ of some equation is established by the sole knowledge of the graph $G(\mathcal{E})$.

3.1. The Algebra of Proofs. To a set of equations $\mathcal{E}$ we associate an algebra $A(\mathcal{E})$, which is a non-commutative non-distributive version of the group ring (over the integers) of the fundamental group of the graph $G(\mathcal{E})$ [34]. The algebra $A(\mathcal{E})$ possesses a binary product, associative with unit 1; and a binary sum. It is freely generated by the (oriented) edges of $G(\mathcal{E})$ and their (formal) inverses. We denote by $s(a)$ (resp. $t(a)$) the vertex source (resp. target) of the generator a. This algebra is naturally endowed with a unary inverse operation, satisfying $(A+B)^{-1} = A^{-1} + B^{-1}$ and $(A.B)^{-1} = B^{-1}.A^{-1}$. Intuitively, expressions in $A(\mathcal{E})$ will denote the flow or current of information in $G(\mathcal{E})$, modelizing deductions in LE under the hypotheses $\mathcal{E}$, or terms built from $\mathcal{E}$, or nothing at all. The product is the concatenation of paths, the sum is the parallel traveling of paths. Taking inverses corresponds to reversing the flow of information, via the symmetry rule (naturally, instances of the reflexivity axiom define an equation of $\mathcal{E}$). A last word on the relation between $\mathcal{E}$ and $A(\mathcal{E})$. Terms in $\mathcal{E}$ represent functions in extension, as usual in the set-theoretic approach to mathematics. Expressions in $A(\mathcal{E})$ correspond to the dual, intensional, view of functions, relevant to Computer Science. Terms denote the result of a computation (in unification a substitution), expressions denote the computation itself.

The definition of members of $A(\mathcal{E})$ that denote proofs or terms needs some partial operations defined on this algebra. Lower case letters denote generators of $A(\mathcal{E})$, upper case letters denote arbitrary expressions. If the term M occurs in $\mathcal{E}$, we define the expression $l(M)$ (resp. $r(M)$) by

- $l(x) = r(x) = 1$, if x is a variable;
- $l(f(M,N)) = l(M).a^{-1} + l(N).b^{-1}$,
  $r(f(M,N)) = a.r(M) + b.r(M)$,
  if a (resp. b) is the edge from $V(f(M,N))$ to $V(M)$ (resp. $V(N)$) in $G(\mathcal{E})$.

The expression $l(M)$ (resp. $r(M)$) denotes the bottom-up ($\bot - \top$) trip from the leaves of M to its root (resp. top-down, $\top - \bot$, from root to leaves). The merge $A|B$ of two expressions A and B is defined as follows (think connection of a plug with a socket):

- $(A.a)|(b.B) = A.(a.b).B$;
- $(A.(A_1 + A_2))|((B_1 + B_2).B) = A.((A_1|B_1) + (A_2|B_2)).B$.

If some expression A denotes a $\bot - \top$ trip of a term, we want its subexpression $A/_l O$ that denote a subterm defined by some occurrence O; and an expression $A\backslash_l O$ that will denote

DEDUCTION	SEMANTICS
$M = N$	$(l(M), e, r(N))$
$S\,\dfrac{M = N}{N = M}$	$\dfrac{(L, E, R)}{(R^{-1}, E^{-1}, L^{-1})}$
$T\,\dfrac{M = N \quad N = O}{M = O}$	$\dfrac{(L_1, E_1, R_1) \quad (L_2, E_2, R_2)}{(L_1, E_1.(R_1 \mid L_2).E_2, R_2)}$
$IL\,\dfrac{M = N \quad C[N] = O}{C[M] = O}$	$\dfrac{(L_1, E_1, R_1) \quad (L_2, E_2, R_2)}{(D_2[L_1.E_1.(R_1 \mid (L_2/_l O_C))], E_2, R_2)}$
$IR\,\dfrac{M = C[N] \quad N = O}{M = C[O]}$	$\dfrac{(L_1, E_1, R_1) \quad (L_2, E_2, R_2)}{(L_1, E_1, D_1[((R_1/_r O_C) \mid L_2).E_2.R_2])}$
$E\,\dfrac{C[M] = D[N]}{M = N}$	$\dfrac{(L, E, R)}{(L/_l O, (L\backslash_l O).E.(R\backslash_r O), R/_r O)}$

the multipath from the root to this subterm (resp. $\top - \bot$, $A/_r O$ and $A\backslash_r O$). Notice the complementary feature of the two definitions.

- $A/_l \epsilon = A$;
- $(a.A_1 + A_2).A/_l 0 = (A_1 + a.A_2).A/_l 1 = 1$;
- $((A_1 + A_2).A_3 + A_4).A/_l 0O = (A_4 + (A_1 + A_2).A_3).A/_l 1O = (A_1 + A_2)/_l O$;
- $a\backslash_l \epsilon = 1$;
- $(a.A_1 + A_2).A\backslash_l 0 = (A_1 + a.A_2).A\backslash_l 1 = a.A_1.A$;
- $((A_1 + A_2).A_3 + A_4).A\backslash_l 0O = (A_4 + (A_1 + A_2).A_3).A\backslash_l 1O = ((A_1 + A_2)\backslash_l O).A_3.A$.

We now define the semantics $[\mathcal{D}]$ of some deduction $\mathcal{D}$. We associate a triple of expressions to a deduction. The semantics is given above, where, in the first row, e is the edge from $V(M)$ to $V(N)$ in $G(\mathcal{E})$; in the fourth and fifth row, the contexts $D_2[\_]$ and $D_1[\_]$ are defined by the identities $L_2 \equiv D_2[L_2/_l O_C]$ and $R_1 \equiv D_1[R_1/_r O_C]$, with $O_C \neq \epsilon$ (i.e. the introductions are true introductions, not mere transitivity rules); finally in the last row the occurrence O is the common occurrence of the holes of the contexts $C[\_]$ and $D[\_]$.

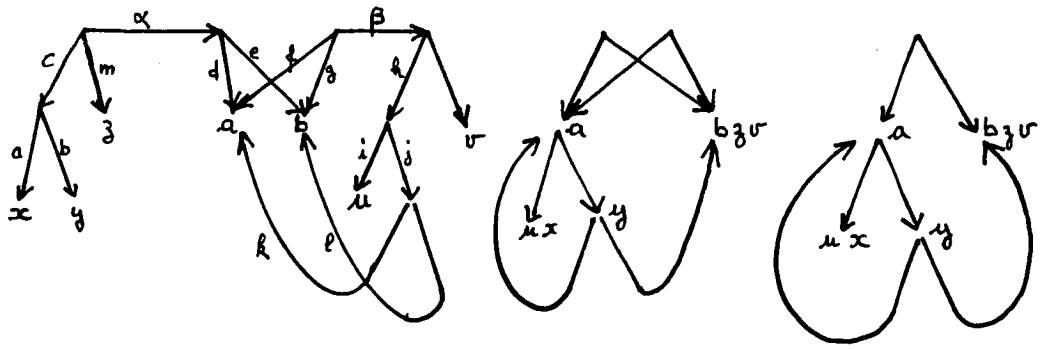

Figure 9: The three graphs $G(\mathcal{E})$, $U(\mathcal{E})$ and $W(\mathcal{E})$.

The central expression of each triple records the computation done in order to establish the equality of the two terms described by the left and right expressions.

Let us see an example. Given $\mathcal{E}$ with two equations $f(f(x,y),z) = f(a,b)$ and $f(a,b) = f(f(u,f(a,b)),v)$, we build the deduction $\mathcal{D}$:

$$IR\ \dfrac{E\ \dfrac{T\ \dfrac{f(f(x,y),z) = f(a,b) \quad f(a,b) = f(f(u,f(a,b)),v)}{f(f(x,y),z) = f(f(u,f(a,b)),v)}}{f(x,y) = f(u,f(a,b))} \qquad S\ \dfrac{f(f(x,y),z) = f(a,b)}{f(a,b) = f(f(x,y),z)}}{f(x,y) = f(u,f(f(x,y),z))}$$

The three $\mathcal{E}$-graphs are given in Fig. 9. The semantics is easily computed, notice its concision as opposed to the redundancies of $\mathcal{D}$:

$$[\mathcal{D}] = (a^{-1} + b^{-1}\ ,\ c^{-1}\alpha(df^{-1} + eg^{-1})\beta h\ ,\ (i + j(kd^{-1} + le^{-1})\alpha^{-1}(c(a+b) + m))).$$

The reader is urged to consider how some usual notions of proof theory receive here their natural expression and to play with these definitions. The elimination E effectively eliminates some information and is the unique rule to do so. The left (right) introduction modifies only the left (right) part of the semantical triple.

Definition 3.1 *A proof is sequential iff all equality components of its subdeductions semantics are sum-free.*

In §3.3, we will analyze some subexpressions of sequential deductions interpretations.

3.2. Normalization. This semantics allows us to find a normalization procedure: two equational deductions are identified iff they have the same semantics. This normalization process is defined by the following rewriting system. We partition the rules according to specific groups. It should be already clear that some of these groups will break the left right symmetry: substitution is a parallel operation, while LE is sequential.

The symmetry group lifts the symmetry rule up to hypotheses. This rule is idempotent and commutes with the remaining rules, which gives five rewriting rules. This group, call

it Sym, is obviously Church-Rosser and well-founded, and trivially interacts with the remaining rules in the rewriting theory sense.

$$Sym \left\{ \begin{array}{lcll} S\dfrac{S\dfrac{M=N}{N=M}}{M=N} & \Longrightarrow & M = N & (1) \\[2ex] S\dfrac{T\dfrac{M=N \quad N=O}{M=O}}{O=M} & \Longrightarrow & T\dfrac{S\dfrac{N=O}{O=N} \quad S\dfrac{M=N}{N=M}}{O=M} & (2) \\[2ex] S\dfrac{IL\dfrac{M=N \quad C[N]=O}{C[M]=O}}{O=C[M]} & \Longrightarrow & IL\dfrac{S\dfrac{C[N]=O}{O=C[N]} \quad S\dfrac{M=N}{N=M}}{O=C[M]} & (3) \\[2ex] S\dfrac{IR\dfrac{M=C[N] \quad N=O}{M=C[O]}}{C[O]=M} & \Longrightarrow & IR\dfrac{S\dfrac{N=O}{O=N} \quad S\dfrac{M=C[N]}{C[N]=M}}{C[O]=M} & (4) \\[2ex] S\dfrac{E\dfrac{C[M]=D[N]}{M=N}}{N=M} & \Longrightarrow & E\dfrac{S\dfrac{C[M]=D[N]}{D[N]=C[M]}}{N=M} & (5) \end{array} \right.$$

The identity group Ide, by rules (6), (7) and (8) below, breaks the left-right symmetry. All contexts are non-trivial.

$$Ide \left\{ \begin{array}{lcll} IL\dfrac{M=N \quad IR\dfrac{C[N]=D[O] \quad O=P}{C[N]=D[P]}}{C[M]=D[P]} & \Longrightarrow & IR\dfrac{IL\dfrac{M=N \quad C[N]=D[O]}{C[M]=D[O]} \quad O=P}{C[M]=D[P]} & (6) \\[2ex] T\dfrac{M=N \quad T\dfrac{N=O \quad O=P}{N=P}}{M=P} & \Longrightarrow & T\dfrac{T\dfrac{M=N \quad N=O}{M=O} \quad O=P}{M=P} & (7) \\[2ex] T\dfrac{IR\dfrac{M=C[N] \quad N=O}{M=C[O]} \quad C[O]=P}{M=P} & \Longrightarrow & T\dfrac{M=C[N] \quad IL\dfrac{N=O \quad C[O]=P}{C[N]=P}}{M=P} & (8) \\[2ex] T\dfrac{IL\dfrac{M=N \quad C[N]=O}{C[M]=O} \quad O=P}{C[M]=P} & \Longrightarrow & IL\dfrac{M=N \quad T\dfrac{C[N]=O \quad O=P}{C[N]=P}}{C[M]=P} & (9) \\[2ex] T\dfrac{M=N \quad IR\dfrac{N=C[O] \quad O=P}{N=C[P]}}{M=C[P]} & \Longrightarrow & IR\dfrac{T\dfrac{M=N \quad N=C[O]}{M=C[O]} \quad O=P}{M=C[P]} & (10) \end{array} \right.$$

The remaining groups are left-right symmetrical. The sequential group Seq contains four

rules.

$$Seq\left\{\begin{array}{l}
IL\dfrac{M{=}N \quad IL\dfrac{C[N]{=}O \quad D[O]{=}P}{DC[N]{=}P}}{DC[M]{=}P} \Longrightarrow IL\dfrac{IL\dfrac{M{=}N \quad C[N]{=}O}{C[M]{=}O} \quad D[O]{=}P}{DC[M]{=}P} \qquad (11)\\[2ex]
IR\dfrac{IR\dfrac{M{=}C[N] \quad N{=}D[O]}{M{=}CD[O]} \quad O{=}P}{M{=}CD[P]} \Longrightarrow IR\dfrac{M{=}C[N] \quad IR\dfrac{N{=}D[O] \quad O{=}P}{N{=}D[P]}}{M{=}CD[P]} \qquad (12)\\[2ex]
IL\dfrac{M{=}C[N] \quad IL\dfrac{N{=}O \quad DC[O]{=}P}{DC[N]{=}P}}{D[M]{=}P} \Longrightarrow IL\dfrac{IR\dfrac{M{=}C[N] \quad N{=}O}{M{=}C[O]} \quad DC[O]{=}P}{D[M]{=}P} \qquad (13)\\[2ex]
IR\dfrac{IR\dfrac{M{=}CD[N] \quad N{=}O}{M{=}CD[O]} \quad D[O]{=}P}{M{=}C[P]} \Longrightarrow IR\dfrac{M{=}CD[N] \quad IL\dfrac{N{=}O \quad D[O]{=}P}{D[N]{=}P}}{M{=}C[P]} \qquad (14)
\end{array}\right.$$

The parallel group Par asserts the identity of two proofs that perform successive unnested or independant introductions. We orientate these equations from left to right, arbitrarily but in accordance with the tradition of leftmost-outermost computation.

$$Par\left\{\begin{array}{l}
IR\dfrac{IR\dfrac{M{=}C[N_1,N_2] \quad N_2{=}O_2}{M{=}C[N_1,O_2]} \quad N_1{=}O_1}{M{=}C[O_1,O_2]} \Longrightarrow IR\dfrac{IR\dfrac{M{=}C[N_1,N_2] \quad N_1{=}O_1}{M{=}C[O_1,N_2]} \quad N_2{=}O_2}{M{=}C[O_1,O_2]} \qquad (15)\\[2ex]
IL\dfrac{M_1{=}N_1 \quad IL\dfrac{M_2{=}N_2 \quad C[N_1,N_2]{=}O}{C[N_1,M_2]{=}O}}{C[M_1,M_2]{=}O} \Longrightarrow IL\dfrac{M_2{=}N_2 \quad IL\dfrac{M_1{=}N_1 \quad C[N_1,N_2]{=}O}{C[M_1,N_2]{=}O}}{C[M_1,M_2]{=}O} \qquad (16)
\end{array}\right.$$

Next, the elimination rule comes into play in the cut-elimination group Cut.

$$Cut\left\{\begin{array}{l}
E\dfrac{IL\dfrac{C[M]{=}N \quad D[N]{=}EF[O]}{DC[M]{=}EF[O]}}{M{=}O} \Longrightarrow E\dfrac{T\dfrac{C[M]{=}N \quad E\dfrac{D[N]{=}EF[O]}{N{=}F[O]}}{C[M]{=}F[O]}}{M{=}O} \qquad (17)\\[2ex]
E\dfrac{IR\dfrac{CD[M]{=}E[N] \quad N{=}F[O]}{CD[M]{=}EF[O]}}{M{=}O} \Longrightarrow E\dfrac{T\dfrac{E\dfrac{CD[M]{=}E[N]}{D[M]{=}N} \quad N{=}F[O]}{D[M]{=}F[O]}}{M{=}O} \qquad (18)\\[2ex]
E\dfrac{IL\dfrac{M{=}N \quad CD[N]{=}E[O]}{CD[M]{=}E[O]}}{D[M]{=}O} \Longrightarrow IL\dfrac{M{=}N \quad E\dfrac{CD[N]{=}E[O]}{D[N]{=}O}}{D[M]{=}O} \qquad (19)\\[2ex]
E\dfrac{IR\dfrac{C[M]{=}DE[N] \quad N{=}O}{C[M]{=}DE[O]}}{M{=}E[O]} \Longrightarrow IR\dfrac{E\dfrac{C[M]{=}DE[N]}{M{=}E[N]} \quad N{=}O}{M{=}E[O]} \qquad (20)
\end{array}\right.$$

In rule (17), $DC[\_]$ is non-trivial, as well as $CD[\_]$ in rule (18), $C[\_]$ in rule (19), $D[\_]$ in rule (20). A phenomenon similar to the short-circuits already noticed by Girard in linear

logic [12] appears in some deductions that forget what they had previously done: some subdeductions are disconnected from the conclusion. Also, the cancellative Can group takes care of stupid deductions.

$$Can \begin{cases} E\dfrac{IL\dfrac{M=N \quad C[N,O]=D[P]}{C[M,O]=D[P]}}{O=P} \Longrightarrow E\dfrac{C[N,O]=D[P]}{O=P} & (21) \\ E\dfrac{IR\dfrac{C[M]=D[N,O] \quad N=P}{C[M]=D[P,O]}}{M=O} \Longrightarrow E\dfrac{C[M]=D[N,O]}{M=O} & (22) \\ E\dfrac{IL\dfrac{M=N \quad C[O,N]=D[P]}{C[O,M]=D[P]}}{O=P} \Longrightarrow E\dfrac{C[O,N]=D[P]}{O=P} & (23) \\ E\dfrac{IR\dfrac{C[M]=D[O,N] \quad N=P}{C[M]=D[O,P]}}{M=O} \Longrightarrow E\dfrac{C[M]=D[O,N]}{M=O} & (24) \end{cases}$$

These twenty-four rules define the rewriting systems:

$$\begin{aligned} RE_S &= Sym \cup Ide \cup Seq, \\ RE_P &= Sym \cup Ide \cup Seq \cup Par, \\ RE_C &= Sym \cup Ide \cup Seq \cup Cut, \\ RE_U &= Sym \cup Ide \cup Seq \cup Par \cup Cut \cup Can. \end{aligned}$$

Theorem 3.1 *The rewriting systems RE_S, RE_P, RE_C and RE_U are strongly normalizing and Church-Rosser.*

Proof. The proof of local confluence is a mechanical critical pair computation. Strong Normalization is established by a lexicographic ordering. First, the number of introduction rules either is constant or decreases under one rewriting step by any rule. This eliminates rules (17), (18), (21) through (24). Next, the sum over introduction rules of the number of (T) or (E) rules under an introduction either is constant or decreases. This eliminates rules (9), (10), (19) and (20). Next, we sum over introduction rules the length of the hole occurrence. This eliminates rules (11) through (14). Next, we count the number of (IL)-rules, thus eliminating rule (8). Rule (6) is ruled out by counting the number of (IL)-rules immediately below (IR)-rules. The termination of the remaining rules (7), (15) and (16) is trivial.□

Practical computations show that there exists only one another *finite* rewriting system for LE, which is obtained by reversing the orientation of the first three rules of the identity group (the orientation of the Par group is unessential). Especially, any orientation of rules (6) or (7), and (8) other than these two ones gives non confluent critical pairs. The system RE_P defines a normalization procedure for (atomic) equational logic. Naturally, we are

interested in deductions in normal form with respect to RE_U. As examples of proofs that could be identified in a truth-valued semantics but with distinct geometrical semantics, we have the following pair, quite characteristic:

$$E\,\dfrac{T\,\dfrac{f(x,y)=f(z,t)\quad f(z,t)=f(f(u,v),w)}{f(x,y)=f(f(u,v),w)}}{x=f(u,v)}$$

$$T\,\dfrac{E\,\dfrac{f(x,y)=f(z,t)}{x=z}\quad E\,\dfrac{f(z,t)=f(f(u,v),w)}{z=f(u,v)}}{x=f(u,v)}$$

As well as the following one:

$$IR\,\dfrac{IR\,\dfrac{f(x,y)=f(z,t)\quad E\,\dfrac{f(z,t)=f(u,v)}{z=u}}{f(x,y)=f(u,t)}\quad E\,\dfrac{f(z,t)=f(u,v)}{t=v}}{f(x,y)=f(u,v)}$$

$$T\,\dfrac{f(x,y)=f(z,t)\quad f(z,t)=f(u,v)}{f(x,y)=f(u,v)}$$

A detailed analysis of the rules gives a description of the normal forms:

1. they are cut-free.
2. No (IR)-rule is right antecedent of an (IL)-rule.
3. If an (IL)-rule with occurrence O_1 is right antecedent of another (IL)-rule with occurrence O_2, then O_1 is the leftmost occurrence of O_1, O_2.
4. If an (IR)-rule with occurrence O_1 is left antecedent of another (IR)-rule with occurrence O_2, then O_1 is the leftmost occurrence of O_1, O_2.
5. If some introduction is antecedent of some (T)-rule, it is an (IR)-rule, which is left antecedent of this (T)-rule.
6. No (T)-rule is right antecedent of another (T)-rule.

Finally, a deduction in normal form does not possess short-circuits. A typical normal form is given below, where we allow for simplicity multiple premisses for binary rules. Let $\mathcal{T}$

be the following deduction:

$$\dfrac{\dfrac{\dfrac{E_n \cdots E_1 D[\overline{M}] = F_n Q^n_{m_n} \qquad \mathcal{F}^n_{m_n} \qquad\qquad \mathcal{F}^n_1}{E_{n-1} \cdots E_1 D[\overline{M}] = Q^n_{m_n}} \; Q^n_{m_n} = Q^n_{m_{n-1}} \cdots Q^n_1 = F_{n-1} Q^{n-1}_{m_{n-1}}}{T}}{E_{n-1} \cdots E_1 D[\overline{M}] = F_{n-1} Q^{n-1}_{m_{n-1}}}$$

$$\vdots$$

$$\dfrac{\dfrac{\dfrac{E_1 D[\overline{M}] = F_1 Q^1_{m_1}}{D[\overline{M}] = Q^1_{m_1}} \qquad \mathcal{F}^1_{m_1} \qquad\qquad \mathcal{F}^1_1}{E} \; Q^1_{m_1} = Q^1_{m_1 - 1} \cdots Q^1_1 = C[\overline{N}]}{T}$$

$$D[M_1, \ldots, M_l] = C[\overline{N}]$$

A normal form:

$$\dfrac{\dfrac{\dfrac{\mathcal{D}_1 \qquad \mathcal{D}_l \qquad\qquad \mathcal{T}}{P_1 = M_1 \cdots P_l = M_l \qquad D[M_1, \ldots, M_l] = C[\overline{N}]}}{IL} \qquad \mathcal{E}_1 \qquad\qquad \mathcal{E}_k}{D[P_1, \ldots, P_l] = C[N_1, \ldots, N_k]} \; N_1 = O_1 \cdots N_k = O_k}{IR}$$

$$D[P_1, \ldots, P_l] = C[O_1, \ldots, O_k]$$

On subdeductions, we have the following restrictions: $\mathcal{D}_i$ and $\mathcal{E}_j$, $i = 1, \ldots, l$, $j = 1, \ldots, k$, are deductions in normal form; $\mathcal{F}^p_q$, $p = 1, \ldots, n$, $j = 1, \ldots, m_p$, are deductions in normal form whose last inference rule is not a (T)-rule nor an (IR)-rule. Especially, we get the usual property of normal forms in sequent calculi [11]: the introductions follow the eliminations. These normal forms also possess an important property, expressed in terms of directions of computations: intuitively, in computing a deduction, subdeductions corresponding to principal premisses should be computed first. Now, the normal forms just formalize this simple idea that in the semantics of a deduction, the equality component should be computed first.

Theorem 3.2 *The rewriting systems RE are conservative with respect to the geometrical semantics:* $\mathcal{D} \Rightarrow^* \mathcal{D}'$ *implies* $[\mathcal{D}] = [\mathcal{D}']$.

Proof. Immediate by induction on deductions. The detailed proof requires the following property, easy consequence of the definition of the merge operation (cf. the semantics of both sides of rule (13)), provided that $E \neq 1$, which is always the case on the semantics of rules: $L_1|(L_2.E.(R_1|R_2)) = ((L_1|L_2).E.R_1)|R_2$ □

As immediate application, Definition 3.1 is coherent with rewriting.

3.3 Operationality. Now, we restrict to sequential fixed-point deductions and explain their operational content via their geometrical interpretation. The correct setting for expressing this operationality is the theory of functional equations (see [1,32]). This analysis is applied to Type Inference for system F. Essentially, we get necessary conditions for a typing scheme of some pure λ-term to give a correct typing of this term in system F. Also, we assume in this section knowledge of typed lambda calculi [3,12]. Our version of the simply typed λ-calculus has one base type, the kind *Type*. We abbreviate the type

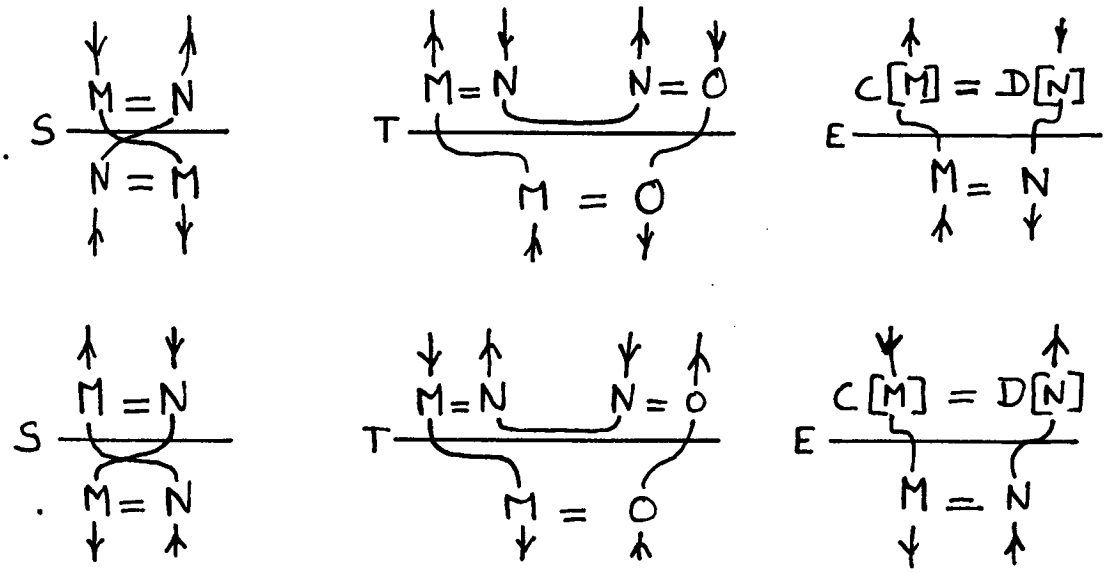

Figure 10: Traveling an inference rule

$Type \to \cdots \to Type$, with $n+1$ instances of $Type$ by $Type^n \to Type$. The constant of the calculus are a binary infix constant $\Rightarrow$ of type $Type \to Type \to Type$, and Church's quantifiers Π_n of type $(Type^n \to Type) \to Type$. The normal form of term M is denoted by $M\downarrow$. Sequences of terms will be conveniently denoted by surligning meta-variables. The rank of a simple type is defined by $rank(Type) = 0$ and $rank(\tau_1 \to \cdots \to \tau_n \to Type) = max(rank(\tau_i)) + 1$. We are given three disjoint sets of variables $\mathcal{T}$, $\mathcal{X}$ and $\mathcal{B}$. Variables in $\mathcal{T}$ or $\mathcal{X}$ are of rank zero or one, variables in $\mathcal{B}$ are or rank zero. Unless specified, λ-terms are in $\beta\eta$-long normal form, η-expansions being done via $\mathcal{B}$-variables. For Type Inference, the intuition is as follows: $\mathcal{T}$-variables represent subterm types, $\mathcal{X}$-variables the type extractions and $\mathcal{B}$-variables the type stratifications, in Girard's terminology [12]. See below for a more usual interpretation in terms of predicate calculi.

If $\mathcal{D} \vdash^{\mathcal{E}}_{LE_1} x = C[x]$ is a sequential fixed-point deduction, with $[\mathcal{D}] = (L, E, R)$, we consider the sum-free expression $\tilde{\mathcal{D}} = L.E.(R/_\tau O_C)$ of $A(\mathcal{E})$, which corresponds to a cycle in $G(\mathcal{E})$. (not a cycle in the sense of §2.3, edges can be traversed in their opposite direction). These deductions and their interpretation have a Question/Response structure, as in Linear Logic [13], which also corresponds to positive/negative occurrences in first-order predicate calculus [8], or to the left/right signatures in first-order classical sequent calculus [11]. If the sequential expression $\tilde{D}$ is "drawn" on a proof, we see an *alternation* of upwards and downwards moves (cf. Fig. 10). Moving upwards can be interpreted as a question, moving downwards as a response. We have inversion of Q/R on axioms, and when identifying the two occurrences of the proper variable of a binary rule. This allows the definition of Positive/Negative or Left/Right occurrences of variables along a deduction. Remember first that to each edge of the graph $G(\mathcal{E})$ corresponds a unique

term occurrence in $\mathcal{E}$.

Definition 3.2 *Let $\mathcal{D} \vdash^{\mathcal{E}}_{LE_1} x = C[x]$ be a sequential fixed-point deduction and $\tilde{\mathcal{D}} = C'[a]$, $a \in G \cup G^{-1}$.*

- *If $s(a) = V(x)$, $x \in \mathcal{V}(\mathcal{E})$, we say that the pair $(a, O_{C'})$ is a left or positive occurrence of x in $\mathcal{D}$, and that x is a marked variable of $\mathcal{D}$.*
- *If $t(a) = V(x)$, $x \in \mathcal{V}(\mathcal{E})$, we say that the pair $(a, O_{C'})$ is a right or negative occurrence of x in $\mathcal{D}$, and that x is a marked variable of $\mathcal{D}$.*

Notice the possibility that some variable occurrence in $\mathcal{E}$, i.e. some generator $a \in G$ such that $s(a) = V(x)$ or $t(a) = V(x)$, $x \in \mathcal{V}(\mathcal{E})$, possesses several positive and/or negative occurrences in a deduction. Some immediate properties (established by easy inductions):

- The left-hand side occurrence of x in a fixed-point sequential deduction $\mathcal{D} \vdash^{\mathcal{E}}_{LE_1} x = C[x]$ is positive, the right-hand side occurrence of x is negative.
- The number of positive occurrences is equal to the number of negative occurrences.
- For each binary inference rule of $\mathcal{D}$, one of the two occurrences of its proper variable is positive, the other one negative.

The intuition behind this definition is the usual meaning of Positivity and Negativity in predicate calculi: positive occurrences contribute to the "strength" of the left-hand side of the conclusion of the deduction $\mathcal{D}$, negative ones to the strength of the right-hand side occurrence of x in the same conclusion. Naturally, Unification Logic, sharing the symmetries of Linear Logic, discovered in an independant context, confirms the accuracy of the intuitions behind these logics.

We explain how all the above material applies to second-order unification, firstly on an example. Consider the following set of (1st-order) equations:

$$\mathcal{E} \begin{cases} x &= f(x,y) \\ x &= f(f(x,z),t) \end{cases}$$

The set $\mathcal{C}(\mathcal{E})$ contains, among other deductions, the hypotheses (each hypothesis defines an elementary cyclic set) and the derivation $\mathcal{D}_3$:

$$u\,\frac{x = f(x,y) \quad x = f(f(x,z),t)}{x = f(x,z)}$$

The positivity/negativity of the (sequential) deduction $\mathcal{D}_3$ is displayed below:

$$\begin{aligned} x^- &= f(x^+,y) \\ x^+ &= f(f(x^-,z),t) \end{aligned}$$

Consider now some associated higher-order equations:

1. $X(A) = f(\Pi(X), y)$, $X(B) = f(f(\Pi(X), z), t)$.
2. $X(A) = f(\Pi(X), y)$, $X(A) = f(f(\Pi(X), z), t)$.
3. $X = \lambda y.f(X(A), y)$, $X = \lambda t.f(f(\Pi(X), z), t)$.
4. $X = \lambda a.f(X(a), y)$, $X(A) = f(f(\Pi(X), z), t)$.

The reader can check that all sets, except 2., protect the deduction $\mathcal{D}_3$, but that only the first set protects simultaneously all elements of $\mathcal{C}(\mathcal{E})$. We now formalize these ideas.

Definition 3.3 *A set $\mathcal{E}$ of higher-order equations between terms of the simply typed λ-calculus over some set of constants is almost first-order iff:*

1. *the set of free variables of $\mathcal{E}$ is included in $\mathcal{T} \cup \mathcal{X}$, (of rank zero or one);*
2. *the set of bound variables of $\mathcal{E}$ is included in $\mathcal{B}$, (of null rank);*
3. *a term is simple iff it is a pure applicative term of type Type built on $\mathcal{X}$-variables and $\mathcal{B}$-variables; arguments of $\mathcal{T}$-variables are simple terms;*
4. *a term is atomic iff it is purely applicative of type Type with head variable a $\mathcal{T}$-variable; a term is compound iff it is*
   - *atomic;*
   - *or is a λ-closure of a compound term;*
   - *or is of the form $C_1 \Rightarrow C_2$, C_1, C_2 compound terms of type Type;*
   - *or is of the form $\Pi_n(C)$, C a compound term of type $Type^n \rightarrow Type$;*
5. *members of equations in $\mathcal{E}$ are compound terms.*

The alert reader will have recognized in this calculus a subsystem of intuitionist first-order predicate calculus (i.e. without negation, conjunction nor disjunction), where $\mathcal{T}$-variables are predicate letters, simple terms are "terms" in this logic, i.e. $\mathcal{X}$-variables correspond to function letters, and finally compound terms denote formulæ. We are trying to solve such equations, that is we want an explicit form defining both the "predicate letters" in $\mathcal{T}$ and the "function symbols" in $\mathcal{X}$. This is done classically with higher-order unification [19]. We now give a necessary condition for such a set of equations $\mathcal{E}$ to have solutions, via fixed-point equations deducible from $\mathcal{E}$. To this end, we *lift* our logical system LE to a second-order equational system LE_2, thus defining *unit resolution* [30,8] for this intuitionistic calculus.

Definition 3.4 *The system LE_2 over compound terms is:*

$$LE_2 \left\{ \begin{array}{lll} R\,\dfrac{}{M = M} & S\,\dfrac{M = N}{N = M} & T\,\dfrac{M = N \quad N = O}{M = O} \\ IL\,\dfrac{M = N \quad C[N] = O}{C[M] = O} & E\,\dfrac{C[M] = D[N] \quad C[\_] \sim D[\_]}{M = N} & IR\,\dfrac{M = C[N] \quad N = O}{M = C[O]} \\ A\,\dfrac{M = N}{(MX)\downarrow = (NX)\downarrow} & & \Lambda\,\dfrac{M = N}{\lambda\alpha.M = \lambda\alpha.N} \end{array} \right.$$

where X is a simple term, α some $\mathcal{B}$-variable, and all proper terms of binary inferences are atomic.

The reader will easily check that this definition is sound, i.e. if the premisses are compound terms, so is the conclusion. A unit resolution step corresponds to two sequences of applications (rule A), whose conclusions are the premisses of some binary rule, followed by some sequence of abstractions (rule Λ). The intuition is as in first-order predicate calculus: we unify the arguments of some $\mathcal{T}$-predicate letter, then send the most general unifier as arguments to λ-terms. After the binary rule, we abstract the variable introduced by unification in order to get the most general conclusion (cf. above for some examples). We are now in position to state our necessary condition:

Definition 3.5 *Let $\overline{X}$ and $\overline{Y}$ be two sequences of simple terms of the same length. We say that $\overline{X}$ dominates $\overline{Y}$, $\overline{X} \gg \overline{Y}$, iff there exists an index i such that, X_i and Y_i being the i^{th} elements of $\overline{X}$ and $\overline{Y}$ respectively:*

- *X_i is not a bound $\mathcal{B}$-variable while Y_i is,*
- *or $X_i = X(\overline{U})$, $Y_i = Y(\overline{V})$ and either X is distinct from Y in $\mathcal{X}$, or $\overline{U} \gg \overline{V}$.*

Some (fixed-point) equation deducible from $\mathcal{E}$ in LE_2 is protected iff it has the form:

$$\lambda\overline{\alpha}.\Phi(\overline{X}) = C[\Phi(\overline{Y})],$$

where $\overline{X} \gg \overline{Y}$.

The intuition behind this definition is that when Φ and the adequate $\mathcal{X}$-variables are projectors, defined by the dominance relation, the equation becomes, e.g.:

$$\lambda\overline{\alpha}.X(\overline{U}) = C[Y(\overline{V})]$$

with $X \not\equiv Y$, which is no longer a fixed-point equation (cf. above for some examples).

Theorem 3.3 *Let $\mathcal{E}$ be some almost first-order set of equations. If some fixed-point equation deducible from $\mathcal{E}$ in LE_2 is non-protected, then $\mathcal{E}$ has no solution.*

Proof. A simple size-of-term argument shows that if some non-protected cyclic equation is derivable from a set of higher-order equations, then this set has no solution at all. □

Hence protecting all fixed-point equations at 2nd-order is a *necessary* condition for the existence of solutions. The reader may play with the above examples and see how the geometrical interpretation of deductions gives their operational semantics by handling *communication* via resolution:

$$u\,\frac{\Lambda\,\dfrac{\lambda a.X(a) = \lambda a.f(X(a),y)}{X(A) = f(X(A),y)} \qquad X(A) = f(f(\Pi(\lambda a.X(a)),z),t)}{X(A) = f(\Pi(\lambda a.X(a)),z)}$$

Here, the arguments of the proper variable X of the inference have been unified, A being a constant and a a variable. Notice that the set 1. protects the deduction $\mathcal{D}_3$ by ununifiability of the arguments A and B (two distinct constants) of the proper variable X of the inference. Returning to the example of the introduction and applying our criterion, $M = (\lambda x.xx)\ (\lambda xy.xyx)$, we can now reject the structures, where X and $Y(\alpha)$ are 2nd-order types and α a type variable:

$$(\lambda x : \Phi.(xX)x)\ (\Lambda\alpha.\lambda x : \Psi(\alpha), y : \Theta(\alpha).xyx),$$

$$(\lambda x : \Phi.xx)\ (\Lambda\alpha.\lambda x : \Psi(\alpha), y : \Theta(\alpha).((xy)Y(\alpha))x),$$

and accept the following ones:

$$(\lambda x : \Phi.(xX)x)\ (\Lambda\alpha.\lambda x : \Psi(\alpha), y : \Theta(\alpha).((xy)Y(\alpha))x),$$

$$(\lambda x : \Phi.(xX)x)\ (\Lambda\alpha.\lambda x : \Psi(\alpha), y : \Theta(\alpha).((xY(\alpha))y)x).$$

We conclude this section with a word on $\mathcal{C}(\mathcal{E})$. The ususal notion of stripping, see e.g. [4], associates to each deduction in LE_2 a deduction in LE, via stripping on compound terms:

- $Strip(\Phi(\overline{X}) = \phi$, $\Phi \in \mathcal{T}$, $\phi \in \mathcal{V}$;
- $Strip(\lambda.C) = Strip(C)$;
- $Strip(C_1 \Rightarrow C_2) = Strip(C_1) \Rightarrow Strip(C_2)$;
- $Strip(\Pi_n(C)) = Strip(C)$.

Thus it is necessary to protect all deductions in $\mathcal{C}(\mathcal{E})$ when they are lifted to second-order. This idea motivated the present paper. Naturally, for a λ-term, stripping just gives its first-order inference set of equations. The justification of the restriction to sequential deductions is clear from the two examples given at the end of §3.2: protecting a sequential deduction is a stronger requirement than protecting a parallel one. Finally, the reader can check that the parallelism of the sum is *synchronous* (or additive in the terminology of Girard [13]). If we drop the restriction on LE_2 that proper terms are atomic, we obtain a non-unit resolution, allowing the lifting of non-sequential first-order deductions in LE. Operationality means here that we have to synchronize several atomic unifications.

4 Separation of Cycles in Unification Graphs

The goal of this section is to establish that a set of equations, non-unifiable in a minimal sense, contains exactly one positive occur-check. This gives first insights about some "generators" of the set of fixed-points deductions. Here and in §5, we work exclusively in the unification graph $U(\mathcal{E})$ of a set $\mathcal{E}$. We establish the existence of "subgraphs" $G(E)$ for $E \subseteq \mathcal{E}$ that separate the cycles in $G(\mathcal{E})$. We use the notations of §2.3, especially the notations relative to its unification algorithm.

Definition 4.1 *The set E' is a one-step linearization of a set E of equations iff E' is obtained from E by renaming some occurrence of a variable that occurs at least twice in E. The set E' is a linearization of E iff there is a non-void sequence of one-step linearizations from E to E'. An elementary cyclic set E is a set of equations such that:*

1. *$G(E)$ contains at least one cycle,*
2. *the graph $G(E')$ is a dag for every linearization E' of E,*
3. *the graph $G(E')$ is a dag for every proper subset E' of E.*

A variable $x \in \mathcal{V}$ is needed iff x possesses at least two distinct occurrences in E. A variable $x \in \mathcal{V}$ is cyclic iff x is needed and $V(x)$ belongs to some cycle. A variable $w \in \mathcal{R}$ is needed iff $val(w)$ contains at least one occurrence of a needed $\mathcal{V}$-variable. A vertex v is needed iff it contains at least one needed $\mathcal{V}$-variable.

If $G(E)$ is a dag, so is $G(E')$ for E' a linearization of E. Linearization stepwise removes the cycles. Let $\mathcal{E}$ be a unification problem. As the powerset of $\mathcal{E}$ is finite and there is a finite number of possible linearizations from a given set, the effectiveness of the base of elementary cyclic sets is trivial. Contrarily to the results of the previous section, the following theorem is fairly insensitive to strict equations. Without loss of generality we may assume that the input E does not contain such equations. The proof proceeds as follows: when the graph has no roots, the result is trivial. Otherwise, we first isolate an equation not embedded in the cycle and associate to it a path that will create a cycle. Then by the results of the previous section, this path is easily shown to be unique. About the technicalities of the proof, remind our constructive option through the paper.

Theorem 4.1 *Let E be an elementary cyclic set of equations, then $G(E)$ contains a unique cycle. Let c be the cycle of $G(E)$, then each vertex of c contains a unique needed $\mathcal{R}$-variable (unique needed non-variable subterm occurrence of E).*

Proof. We consider two cases: either $G(E)$ contains some root or it does not. In the latter case, we establish that each equation in E possesses exactly one marked occurrence in its (non-strict) right-hand side. We select a non $\mathcal{V}$-free vertex v_0. By Lemma 2.5, there exists some variable x_0 in v_0 that occurs in some non-strict right-hand side of an equation e_0.

Let v_1 be the vertex of the left-hand side x_1 of this equation. As we do not have strict equations, by the same Lemma the variable x_1 occurs in some non-strict right-hand side of equation e_1, and so on. The number of equations being finite, let j, n be the smallest integers so that $e_j = e_{j+n}$. This subsequence defines a cycle. By definition of E, each equation occurs exactly once in this subsequence and each left-hand side has exactly two occurrences: one as left-hand side of e_i and one in the right-hand side of e_{i+1}. No other variable occurs twice in E, and this cycle is obviously the only one in $G(E)$.

In the former case, for any cycle, there exists some cyclic vertex that possesses at least two incident edges. Such vertices are non $\mathcal{V}$-free by Proposition 2.6. There exists at least one such vertex v such that some $\mathcal{V}$-variable $y \in v$ occurs by an edge not belonging to any cycle. To see this, let v be some root. The set E being elementary, there exists at least one path from v to some cyclic vertex. Let $p = v, \ldots, v'$ be such a path, minimal in the sense that for no subpath $p' = v, \ldots, v''$ of p we have v'' cyclic. Then v' is shared and is the required vertex. Let y be a cyclic variable in this vertex, so that y occurs by an edge not belonging to the cycle. This occurrence belongs to the right-hand side of an equation $e : x = C[y]$. We may further assume that y possesses at least two distinct occurrences by Proposition 2.6. Without loss of generality we assume that the equation e has a single variable occurrence in its right-hand side whose variable possesses multiple occurrences, namely y. If not so the equation e can be replaced by two equations without altering the graphs of the set E. The resulting set is still elementary cyclic. Such an equation will be said linearized. By definition of an elementary cyclic set, both graphs $G(E - \{e\})$ and $G(E')$ are dags, where E' is obtained from E by linearizing the occurrence O_C. We assume that the equation e is the last level 0 equation to be selected, say at iteration n.

Finally, concerning the existence of cycles in the graphs, we note that the graph G^{i+1} is cyclic while all G^j are dags for $j \leq i$ iff there exists some path $V^i(w_1^i), \ldots, V^i(w_2^i)$ or $V^i(w_2^i), \ldots, V^i(w_1^i)$. In order to complete the proof, we need four technical lemmas on the existence of paths in the graphs G_i. Intuitively, under our choice of the equation $x = C[y]$, the cycle will be created while merging two vertices, one in $G^n \downarrow V^n(x)$, the other in the term dag of $C[y]$. Our intuition also tells us that this path already exists in G^n. In order to formalize this, we need to pull some paths in G^j back to G^i, $i < j$.

Lemma 4.2 *Let $i < j < i^+$ be three iterations. There exists two paths $p_1 = V^i(w_k^i), \ldots, V^i(w_1^j)$, $k = 1$ or 2, and $p_2 = V^i(w_l^i), \ldots, V^i(w_2^j)$, $l = 1$ or 2.*

Proof. By induction on $n = j - i$. If $n = 1$ the equation selected at iteration j has been created at iteration i. The two paths are defined by the function *succ*.

Assume the lemma true for $m \leq n - 1$. Let $w_1^j = w_2^j$ be the equation selected at iteration j such that $j - i = n$. This equation has been created at some iteration k_0 with $i \leq k_0 < j$. Without loss of generality we assume that this equation is the left one. By induction hypothesis, we have two paths p'_l ending in $V^i(w_l^{k_0})$, $l = 1, 2$. At iteration k_0 these two vertices possess successors. If each vertex $V^i(w_l^{k_0})$ also possesses a left successor

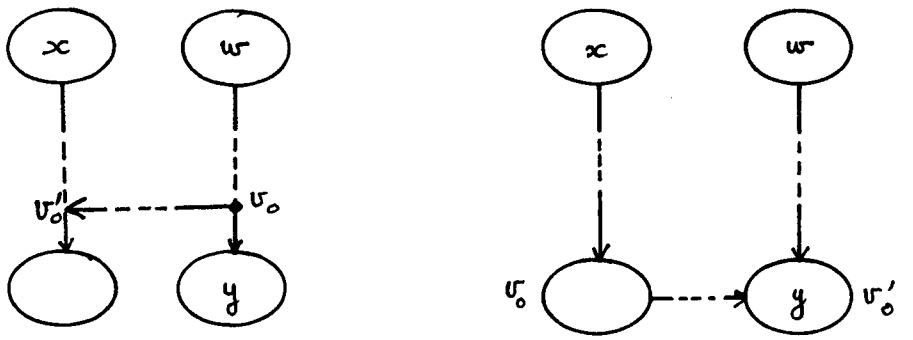

Figure 11: Paths in Lemma 3.4.

that include the variable w_l^j, the result is immediate. Otherwise, one of these vertices, say $V^i(w_1^{k_0})$ does not possess w_1^j in its (possibly non-existent) left successor. Let k_1, $k_1 < k_0$, be the first iteration such that w_1^j is in the left successor of $V^{k_1+1}(w_1^{k_0})$. We apply the induction hypothesis to iteration k_1 and consider the path ending in the vertex $V^i(w_l^{k_1})$, included in the vertex $V^{k_1}(w_l^{k_1})$, $l = 1$ or 2, that does not contain $w_1^{k_0}$ but possesses two successors, the left one containing w_1^j. If $V^i(w_l^{k_1})$ possesses a left successor containing w_1^j, we get the result. Otherwise we iterate the construction by considering the first iteration k_2, $k_2 < k_1$, such that $V^{k_2+1}(w_l^{k_1})$ contains w_1^j in its left successor. This halts as $i \leq \cdots < k_1 < k_0 < j$. □

Lemma 4.3 *Let $i < j < i^+$ be three iterations. If there exists a path $p_j = V^j(w_k^j), \ldots, V^j(w)$, $k = 1$ or 2, there exists a path $p_i = V^i(w_l^i), \ldots, V^i(w)$, $l = 1$ or 2.*

Proof. By Lemma 4.2 there exists a path $p_1 = V^i(w_l^i), \ldots, V^i(w_k^j)$, $l = 1$ or 2. If there exists a path $p_2 = V^i(w_k^j), \ldots, V^i(w)$, we take p_i to be the concatenation of p_1 and p_2. Otherwise, let k_0, $i < k_0 < j$ be the first iteration so that there exists a path $V^{k_0+1}(w_k^j), \ldots, V^{k_0+1}(w)$. By Lemma 4.2, there exists a path $p_1 = V^i(w_l^i), \ldots, V^i(w_m^{k_0})$, $m = 1$ or 2, together with a path $V^{k_0}(w_m^{k_0}), \ldots, V^{k_0}(w)$. If there exists a path $p_2 = V^i(w_m^{k_0}), \ldots, V^i(w)$, we take p_i to be the concatenation of p_1 and p_2. Otherwise we iterate the construction by considering the first iteration k_1, $k_1 < k_0$, such that there exists a path from $V^{k_1+1}(w_m^{k_0})$ to $V^{k_1+1}(w)$. This halts as $i \leq \cdots < k_1 < k_0 < j$. □

The next lemma establishes the existence of the path mentionned before Theorem 4.1.

Lemma 4.4 *Let $e : x = C[y]$ be as in the proof of Theorem 4.1, then there exists an occurrence O, prefix of O_C, and a path $p = v_0, \ldots, v_0'$ in G^n, n the iteration that selects e, with $v_0 = V^n(w)/O$ and $v_0' = V^n(x)/O$, or $v_0 = V^n(x)/O$ and $v_0' = V^n(w)/O$, where $w \in \mathcal{R}$ is associated to the (non-strict) right-hand side of e.*

Proof. Cf. Fig. 11. Assume the lemma false. The equation e being linearized, the term dag representing $val(w)$ is a tree, where $V^n(y)$ is the only leaf that can possess successors

belonging to G^n. At iterations following the n^{th} one, we assume that equations not along O_C in the tree $G \downarrow V^n(w)$ are first selected. By hypothesis, $V^N(y)$ is cyclic, while G^n is a dag. If the vertex $V^n(x)/O_C$ is undefined, or if one of the vertices $V^n(x)/O_C$ and $V^n(y) = V^n(w)/O_C$ does not possess successors, then G^N is still acyclic. This is so as 1) the occurrence of y in e is needed in the elementary cyclic set E, 2) the lemma is assumed to be false and 3) e is linearized.

Hence, let i be the iteration that selects $w_1^i = w_2^i$ such that $i > n$ and no equation selected at iteration k, $n < k < i$, satisfies $V^k(w_1^k) = V^k(w')$ and $V^k(w_2^k) = V^k(y)$, w' some variable in $V^n(x)/O_C$. We have $V^i(w_1^i) = V^i(w')$ and $V^i(w_2^i) = V^i(y)$. By our assumption, G^{i+1} is acyclic. Also, let j, $i < j < i^+ = N$ be the first iteration so that $V^{j+1}(y)$ is cyclic. There exists a path p between the vertices of w_1^j and w_2^j, e.g. $p = V^j(w_1^j), \ldots, V^j(w_2^j)$, such that $V^j(y) \in p$ (otherwise, by minimality of E, $V^N(y)$ would not be cyclic). By Lemma 4.3 we have a path $p_i = V^i(w_l^i), \ldots, V^i(y)$. Necessarily $w_l^i = w_1^i$ as G^i is acyclic. As equation e is linearized, p_i belongs to G^n, contradiction. □

Lemma 4.5 *Let O be the smallest occurrence satisfying the conditions of Lemma 4.4, then there exists a unique path from v_0 to v_0'.*

Proof. First of all, notice that there does not exist a path from v_0' to v_0 as G^n is acyclic. Further, notice that there does not exist a path from $V^n(y)$ to $V^n(x)$ as $V^N(x)$ is a root. Consequently, $|O| > 0$. Let p_1 be a path from v_0 to v_0'. By O's minimality, the edge of p_1 incident to v_0' is distinct from the edge incident to v_0' along either the path $V^n(x), \ldots, v_0'$ or $V^n(w), \ldots, v_0'$, i.e. along O_C. Otherwise, we have a path between the predecessors of v_0 and v_0' along O_C, which contradicts O's minimality. Therefore, v_0' possesses two distinct incident edges and is non $\mathcal{V}$-free by Proposition 2.6. Let p_2 be another path from v_0 to v_0'. Let v be the first vertex above v_0' so that the path $p_3 = v, \ldots, v_0'$ is the maximal suffix path common to both p_1 and p_2. Then v possesses two distinct incident edges e_1 and e_2, $e_1 \in p_1$, $e_2 \in p_2$. Let $x_0 \in \mathcal{V}_{v_0'}$ that occurs by the common path p_3. Let $e_0 : y_0 = C_0[x_0]$ be an associated equation, i.e. an equation such that the edge incident to $V^0(x_0)$ in the term dag of $C_0[x_0]$ is congruent to the last edge of p_3. We have two cases: either the path $V^n(y_0), \ldots, V^n(x_0)$ along O_{C_0} includes p_3 or it does not. In the second case, by Proposition 2.6 and Lemma 2.5, true at level 0 iterations, we have a sequence of multiple occurring variables starting with y_0 and ending with a variable x_1 that occurs by p_3. Hence this second case ends up in the first one after a sequence of multiple occurring variables along p_3, cf. Fig. 12.

The last equation so constructed $y_m = C_m[x_m]$ defines a path $V^n(y_m), \ldots, V^n(x_m)$ either (i) including one of e_1 or e_2 or (ii) $V^n(y_m) = v$ or (iii) v possesses three incident edges, the third one e_3 being associated to the path $V^n(y_m), \ldots, V^n(x_m)$ defined by the last equation. In any case, one variable occurrence, occurring either by e_1 or e_2, can be linearized without destroying one of the two paths p_1 or p_2. The resulting set is still cyclic as one path, say p_1, always exists in G^n.

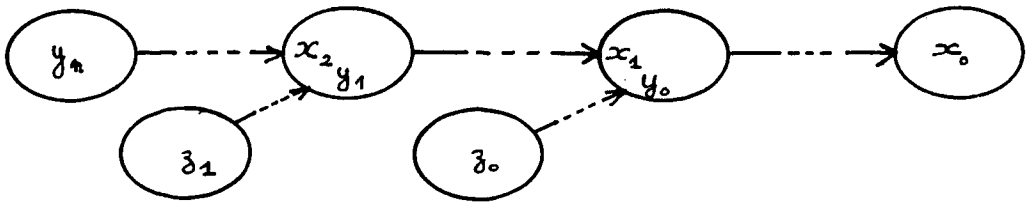

Figure 12: Sequence of equations and chains on a path.

In both cases we get a contradiction with E an elementary cyclic set. Hence there exists a unique path from v_0 to v_0'. □

We conclude the proof of Theorem 4.1. By the previous Lemmas, there exists an occurrence O such that we have a unique path between $V^n(x)/O$ and $V^n(w)/O$, say from v_0 to v_0'. The two graphs $G^n \downarrow v_0$ and $G^n \downarrow v_0'$ are trees. Otherwise they are dags such that at least one vertex, distinct from both v_0 and v_0', has multiple predecessors. Linearizing one variable in this vertex does not modify the unique path between v_0 and v_0', contradiction. The same observation implies that the trees are disjoint. Therefore $G(E)$ contains a unique cycle as the iteration that merges v_0 and v_0' creates one cycle, and afterwards two disjoint trees are merged. This does not create any new cycle.

The second part of the Theorem follows from the unicity of the path $p = v_0, \ldots, v_0'$. We first build a chain of needed variables as follows. As v_0' is shared, we may choose some needed $\mathcal{V}$-variable x_0 that occurs in v_0' by the last edge e_0 of p by Proposition 2.6. There exists an equation $e^0 : y_0 = C_0[x_0]$ such that the last edge of the path $p_0 = (V(y_0), O_{C_0})$ is e_0. Let V_1 be the source of the maximal common suffix path of p and p_0. Either $V(y_0) = V_1$ or not, in this latter case V_1 is shared. In both cases V_1 contains at least one needed variable and we can iterate the process with (y_1, x_1) by Proposition 2.6, first part. This stops as soon as V_n is above v_0. The sequence selects for each vertex in p a single needed $\mathcal{R}$-variable. The sequence (y_i, x_i) so constructed is rigid in the sense that there exists a chain as in Proposition 2.6 that links y_{i-1} to x_i.

Assume that some vertex in p contains two distinct needed $\mathcal{R}$-variables w_1 and w_2. One of them, say w_1, is not captured by the above sequence. Then the needed occurrence of some $\mathcal{V}$-variable in $val(w_1)$ is necessarily the occurrence of a variable z such that $V(z) \in p$ by the unicity of the cycle in G^N. Linearizing this occurrence of z preserves the sequence (y_i, x_i) hence the unique path p, and finally the resulting graph is still cyclic, contradiction. □

Hence we have established that every unification problem contains a finite basis of elementary cyclic sets. The unicity property established in the second part of the theorem will be essential in the last section. The proof of Theorem 4.1 gives some hints on a proof-search procedure for a given path. Such a procedure will build a sequence of equations and chains as in Fig. 12, and will direct its search according to the needed $\mathcal{R}$-variables. This intuition lies behind the algorithm of §5.

5 Sequentiality of Elementary Cyclic Sets

We now address the problem of effectively finding an equational deduction of an equation $x = C[x]$ given an elementary cyclic set E so that its cycle is equal to $(V(x), O_C)$. From now on, all deductions are assumed to be in normal form. And we use exclusively the system LE_1. Also the normal forms are those described in the proof of Proposition 2.10. We introduce a new notion of minimality, based on the number of inference rules in a proof, not counting the number of symmetry rules. Also, minimal means that this number is minimum among deductions of some path or cycle. That is we allow the comparison of two deductions $\mathcal{D} \vdash x = C[x]$ and $\mathcal{D}' \vdash y = C'[y]$ provided that the two paths $(V(x), O_C)$ and $(V(y), O_{C'})$ define the cycle of an elementary cyclic set. As opposed to the previous reductions that were local, we need here to have a more global and a more semantical analysis of the deductions. Also, the results are more intricate than those of the previous section. We first establish that in such a minimal deduction, all *marked* variables are needed and the auxiliary deductions do not eliminate cyclic variables. Next, we prove unicity properties for chains in vertices of the graph $G(E)$: we have already proved their existence in Proposition 2.6, given two co-incident edges. For an elementary cyclic set, we have stronger unicity results. These two results imply the correctness of a deterministic algorithm finding a minimal deduction. In turn the existence of this algorithm proves that, up to permutation relative to the cycle, there exists a minimum deduction. We use the notations of §2.3.

5.1. Variable Occurrences in Minimal Deduction. We first establish that a minimal deduction does not involve unneeded variables. The left-hand side of the conclusion of a cyclic inference is its main variable. Remind that the variables that are eliminated by substitutions are its proper variables. Building on example $\mathcal{E}_2$, for the set of equations $\mathcal{E}_5$ below, the following deduction $\mathcal{D}_3$, while in normal form, is surely not the more direct one from a semantical point of view. Hence the following Lemmas have a stronger content than those in the previous section, relatively straighforward.

$$\mathcal{E}_5 \quad \begin{cases} x = f(y,u) \\ x = f(f(u,r),s) \\ x = f(f(v,t),f(v,w)) \end{cases}$$

Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be the following two deductions:

$$(d)\ \dfrac{(d)\ \dfrac{x = f(y,u) \quad x = f(f(v,s),f(v,w))}{y = f(v,s)} \qquad (d)\ \dfrac{x = f(y,z) \quad x = f(f(u,r),s)}{y = f(u,r)}}{v = u}$$

$$(t)\ \dfrac{(d)\ \dfrac{x = f(y,u) \quad x = f(f(u,t),r)}{u = r} \qquad (d)\ \dfrac{x = f(f(u,t),r) \quad x = f(f(v,s),f(v,w))}{r = f(v,w)}}{u = f(v,w)}$$

The following inference $\mathcal{D}_3$ proves the existence of a cycle:

$$(t)\ \dfrac{\overset{\mathcal{D}_1}{v = u} \qquad \overset{\mathcal{D}_2}{u = f(v,w)}}{v = f(v,w)}$$

Naturally, the marked occurrences of y and r in this deduction are not really needed. The present section formalizes this idea for elementary cyclic sets.

Proposition 5.1 *Let $\mathcal{D}$ be a minimal cyclic deduction from an elementary cyclic set E. All marked variables of $\mathcal{D}$ are needed variables of E.*

Proof. Firstly, any left-hand side is marked in $\mathcal{D}$. Otherwise, some equation is redundant, which contradicts E an elementary cyclic set. Secondly, we will prove that if the variable x is proper for some inference rule in $\mathcal{D}$ and the two occurrences of x in the premisses are equal occurrences in some right-hand side of E, then $\mathcal{D}$ is not minimal. This establishes the result but for the main variable of $\mathcal{D}$ (the same question for left-hand sides is trivial). This in turn follows by the same technique. These claims are proved in the three following (technical) lemmas. □

The intuitive idea underlying these lemmas is that if the two occurrences of the proper variable of an inference rule are equal, then we can have a deduction of the conclusion of this inference that does not proceed from these occurrences. This is obtained via the following Lemma. For an example, keep in mind the deduction $\mathcal{D}_3$ for $\mathcal{E}_5$. Recall that $C[\_] \leq D[\_]$ for contexts means that O_C is a prefix of O_D and that when a deduction is in normal form, if some right-hand side occurrence in an hypothesis e is eliminated, it must be extracted from this right-hand side by a subdeduction whose rightmost hypothesis is e, and whose rightmost branch consists of (d)-rules.

Lemma 5.2 *Let $\mathcal{D}$ and $\mathcal{E}$ be the two following deductions:*

$$(d)\ \dfrac{\overset{\mathcal{D}_2}{z_1 = C_2^1[z_2]} \qquad (d)\ \dfrac{\overset{\mathcal{D}_1}{z_0 = C_1^1[z_1]} \quad \overset{\mathcal{D}_0}{z_0 = C_1 \cdots C_k[t]}}{z_1 = C_2 \cdots C_k[t]}}{z_2 = C_3 \cdots C_k[t]}$$

$$\vdots$$

$$(d)\ \dfrac{\overset{\mathcal{D}_k}{z_{k-1} = C_k^1[z_k]} \qquad\qquad z_{k-1} = C_k[t]}{z_k = t}$$

$$(d)\dfrac{\dfrac{\mathcal{E}_2}{y_1 = D_2^1[y_2]}\quad (d)\dfrac{\dfrac{\mathcal{E}_1}{y_0 = D_1^1[y_1]}\quad \dfrac{\mathcal{E}_0}{y_0 = D_1\cdots D_l[s]}}{y_1 = D_2\cdots D_l[s]}}{y_2 = D_3\cdots D_l[s]}$$

$$(d)\dfrac{\dfrac{\mathcal{E}_l}{y_{l-1} = D_l^1[y_l]}\qquad \dfrac{\vdots}{y_{l-1} = D_l[s]}}{y_l = s}$$

such that the two equations $y_0 = D_1\cdots D_l[s]$ *and* $z_0 = C_1\cdots C_k[t]$ *are the same hypothesis (up to strict equality of the left-hand sides). Assume that* $D_1\cdots D_l \leq C_1\cdots C_k$. *There exists a deduction* $\mathcal{D}\wedge\mathcal{E}\vdash y_l = C_i^3[z_i]$, *where* i *is defined by* $C_1\cdots C_{i-1} \leq D_1\cdots D_l \leq C_1\cdots C_i$ *and* $C_i^1 = C_i^2C_i^3$ *by* $C_1\cdots C_{i-1}C_i^2 \sim D_1\cdots D_l$. *Further this deduction does not contain* $\mathcal{D}_0$ *nor* $\mathcal{E}_0$ *as subdeductions.*

Proof. Without loss of generality, we assume that $y_0 \equiv z_0$ and that both $\mathcal{D}_0$ and $\mathcal{E}_0$ are hypotheses (in the general case we have an hypothesis $x = D_1\cdots D_l[s] = C_1\cdots C_k[t]$ and $\mathcal{D}_0$, $\mathcal{E}_0$ decompose in (t)-deductions $z_0 = x$ and $y_0 = x$, completed by this hypothesis). The proof is by induction on l.

For $l = 1$, we have $D_1^1 = E_1\cdots E_i$ with $E_j \sim C_j^1$, $j = 1,\ldots,i-1$ and $C_i^1 = C_i^2C_i^3$ with $C_i^2 \sim E_i$. We build the deduction:

$$(d)\dfrac{\dfrac{\mathcal{D}_2}{z_1 = C_2^1[z_2]}\quad (d)\dfrac{\dfrac{\mathcal{D}_1}{z_0 = C_1^1[z_1]}\quad \dfrac{\mathcal{E}_1}{y_0 = E_1\cdots E_i[y_1]}}{z_1 = E_2\cdots E_i[y_1]}}{z_2 = E_3\cdots E_i[y_1]}$$

$$(d)\dfrac{\dfrac{\dfrac{\mathcal{D}_{i-1}}{z_{i-2} = C_{i-1}^1[z_{i-1}]}\qquad \dfrac{\vdots}{z_{i-2} = E_{i-1}E_i[y_1]}}{z_{i-1} = E_i[y_1]}\qquad \dfrac{\mathcal{D}_i}{z_{i-1} = C_i^2C_i^3[z_i]}}{y_1 = C_1^3[z_i]}$$

Assuming the lemma true for $l-1$, let i be such that $C_1\cdots C_{i-1} \leq D_1\cdots D_{l-1} \leq C_1\cdots C_i$. By induction hypothesis, we have $C_i^1 = C_i^2C_i^3$ and a deduction $\mathcal{D}'\vdash y_{l-1} = C_i^3[z_i]$ without using $\mathcal{D}_0$ nor $\mathcal{E}_0$. There exists j such that $C_1\cdots C_{j-1} \leq D_1\cdots D_l \leq C_1\cdots C_j$. We have $D_l^1 = E_0\cdots E_{j-i}$ with $E_0 \sim C_i^3$, $E_k \sim C_{i+k}^1$, $k = 1,\ldots j-i-1$, and $C_j^i = C_j^2C_j^3$ with $C_j^2 \sim E_{j-i}$. We build the deduction:

$$(d)\dfrac{\dfrac{\mathcal{D}'}{y_{l-1} = C_i^3[z_i]}\quad \dfrac{\mathcal{E}_l}{y_{l-1} = E_0\cdots E_{j-i}[y_l]}}{z_i = E_2\cdots E_{j-i}[y_l]}$$

$$(d)\dfrac{\dfrac{\dfrac{\mathcal{D}_{j-1}}{z_{j-2} = C_{j-1}^1[z_{j-1}]}\qquad \dfrac{\vdots}{z_{j-2} = E_{j-i-1}E_{j-i}[y_l]}}{z_{j-1} = E_{j-i}[y_l]}\qquad \dfrac{\mathcal{D}_j}{z_{j-1} = C_j^2C_j^3[z_j]}}{y_l = C_j^3[z_j]}$$

The reader may check that the presence of (t)-deductions merely complicates the above argument. □

Lemma 5.3 *Let E be an elementary cyclic set. Assume that in some cyclic deduction $\mathcal{D}$, a rule eliminates the variable x and that the two occurrences of x in the premisses are equal occurrences in some right-hand side of E. Then the deduction $\mathcal{D}$ is not minimal among cyclic deductions for E.*

Proof. Given such a deduction, we reduce it according to the kind of the eliminating rule. In each case we have two left (d)-branches, defining two subdeductions $\mathcal{D}$ and $\mathcal{E}$ as in Lemma 5.2. Also, we use this Lemma to get a deduction $\mathcal{D} \wedge \mathcal{E} \vdash y_l = C_j^3[z_j]$. The notation of this Lemma are intensively used in the proof below.

Case of (d)-rule. Without loss of generality the deduction is:

$$(d)\ \dfrac{(d)\ \dfrac{\overset{\mathcal{D}}{z_k = E_1[x]} \quad \overset{\mathcal{D}_1}{z_k = E_2E_3[y]}}{x = E_3[y]} \quad (d)\ \dfrac{\overset{\mathcal{E}}{y_l = E_4[x]} \quad \overset{\mathcal{E}_1}{y_l = E_5E_6E_7[t]}}{x = E_6E_7[t]}}{y = E_7[t]}$$

We have the following equivalences of contexts, as both $E_1[x]$ and $E_4[x]$ come from the same hypothesis by assumption (with the notations of Lemma 5.2):

$$D_1 \cdots D_l E_4 = C_1 \cdots C_k E_1.$$

But $D_1 \cdots D_l = C_1 \cdots C_{j-1} C_j^2$, $C_j^2 C_j^3 = C_j$, hence $E_4 \sim E_5 \sim C_j^3 C_{j+1} \cdots C_k E_1$. With abuse of notations with respect to contexts (we assume that $E_4 \equiv C_j^3 C_{j+1} \cdots C_k E_1$), we build the deduction:

$$(d)\ \dfrac{\overset{\mathcal{D}_1}{z_k = E_2E_3[y]} \quad (d)\ \dfrac{\overset{\mathcal{D}_k}{z_{k-1} = C_k^1[z_k]} \quad \begin{array}{c}\vdots\\ z_{k-1} = C_kE_1E_6E_7[t]\end{array}}{z_k = E_1E_6E_7[t]}}{y = E_7[t]}$$

$$(d)\ \dfrac{\overset{\mathcal{D}_{j+1}}{z_j = C_{j+1}^1[z_{j+1}]} \quad (d)\ \dfrac{\overset{\mathcal{D} \wedge \mathcal{E}}{y_l = C_j^3[z_j]} \quad \overset{\mathcal{E}_1}{y_l = E_5E_6E_7[t]}}{z_j = C_{j+1} \cdots C_kE_1E_6E_7[t]}}{z_{j+1} = C_{j+2} \cdots C_kE_1E_6E_7[t]}$$

Notice that this deduction is strictly smaller than the original one.

Case of (t)-rule. Without loss of generality we have the configuration:

$$(t)\ \dfrac{(d)\ \dfrac{\overset{\mathcal{D}_1}{z_k = E_2[y]} \quad \overset{\mathcal{D}}{z_k = E_1[x]}}{y = x} \quad (d)\ \dfrac{\overset{\mathcal{E}}{y_l = E_4[x]} \quad \overset{\mathcal{E}_1}{y_l = E_3[t]}}{x = t}}{y = t}$$

We have $\mathcal{D} \wedge \mathcal{E} \vdash y_l = C_j^3[z_j]$ and $E_3 \sim C_j^3 C_{j+1} \cdots C_k E_1$. With the same abuse of notations as above, we build:

$$\dfrac{\dfrac{\dfrac{\mathcal{D}_1}{z_k = E_2[y]} \quad (d)\ \dfrac{\dfrac{\mathcal{D}_k}{z_{k-1} = C_k^1[z_k]} \quad \cdots \quad \dfrac{\mathcal{D}_{j+1}}{z_j = C_{j+1}^1[z_{j+1}]} \quad (d)\ \dfrac{\dfrac{\mathcal{D} \wedge \mathcal{E}}{y_l = C_j^3[z_j]} \quad \dfrac{\mathcal{E}_1}{y_l = E_3[t]}}{z_j = C_{j+1} \cdots C_k E_1[t]}}{z_{j+1} = C_{j+2} \cdots C_k E_1[t]} \quad \cdots \quad \dfrac{}{z_{k-1} = C_k E_1[t]}}{z_k = E_1[t]}}{y = t}$$

Case of (su)-rule. We have four subcases. First,

$$\dfrac{\dfrac{\dfrac{\mathcal{D}}{z_k = E_1[x]} \quad \dfrac{\mathcal{D}_1}{z_k = E_2[t]}}{x = t} \quad \dfrac{\mathcal{E}}{y_l = E_4[x]}}{y_l = E_4[t]}$$

With $\mathcal{D} \wedge \mathcal{E} \vdash y_l = C_j^3[z_j]$ and $E_4 \sim C_j^3 C_{j+1} \cdots C_k E_2$, we build:

$$\dfrac{\dfrac{\dfrac{\mathcal{D}_1}{z_k = E_2[t]} \quad (su)\ \dfrac{\dfrac{\mathcal{D}_k}{z_{k-1} = C_k^1[z_k]} \quad \cdots \quad \dfrac{\mathcal{D}_{j+1}}{z_j = C_{j+1}^1[z_{j+1}]} \quad \dfrac{\mathcal{D} \wedge \mathcal{E}}{y_l = C_j^3[z_j]}}{y_l = C_j^3 C_{j+1}^1[z_{j+1}]}}{y_l = C_j^3 C_{j+1}^1 \cdots C_{k-1}^1[z_{k-1}]} \quad \cdots \quad \dfrac{}{y_l = C_j^3 C_{j+1}^1 \cdots C_k^1[z_k]}}{y_l = C_j^3 C_{j+1}^1 \cdots C_k^1 E_2[t]}$$

Notice that the conclusion is not preserved. But we only need the existence of a smaller deduction $\mathcal{D} \vdash y_l = C[t]$ with $C \sim E_4$. That the new deduction is smaller is immediate. Second,

$$\dfrac{\dfrac{\dfrac{\mathcal{E}}{y_l = E_4[x]} \quad \dfrac{\mathcal{E}_1}{y_l = E_3[t]}}{x = t} \quad \dfrac{\mathcal{D}}{z_k = E_1[x]}}{z_k = E_1[t]}$$

With $\mathcal{D} \wedge \mathcal{E} \vdash y_l = C_j^3[z_j]$ and $E_3 \sim C_j^3 C_{j+1} \cdots C_k E_1$, we build

$$\dfrac{\dfrac{\dfrac{\mathcal{D}_{j+1}}{z_j = C_{j+1}^1[z_{j+1}]} \quad (d)\ \dfrac{\dfrac{\mathcal{D} \wedge \mathcal{E}}{y_l = C_j^3[z_j]} \quad \dfrac{\mathcal{E}_1}{y_l = E_3[t]}}{z_j = C_{j+1} \cdots C_k E_1[t]}}{z_{j+1} = C_{j+2} \cdots C_k E_1[t]} \quad \cdots}{z_k = E_1[t]}$$

Notice that here too, the conclusion is only preserved up to equivalence of contexts. Third,

$$(su)\dfrac{(d)\dfrac{\begin{matrix}\mathcal{D}\\ z_k = E_1[x]\end{matrix}\quad\begin{matrix}\mathcal{D}_1\\ z_k = E_2[t]\end{matrix}}{x = t}\quad(su)\dfrac{\begin{matrix}\mathcal{E}\\ y_l = E_4[x]\end{matrix}\quad\begin{matrix}\mathcal{E}_1\\ y = E_3[y_l]\end{matrix}}{y = E_3E_4[x]}}{y = E_3E_4[t]}$$

With $\mathcal{D} \wedge \mathcal{E} \vdash y_l = C_j^3[z_j]$ and $E_4 \sim C_j^3 C_{j+1} \cdots C_k E_2$, we build

$$(su)\dfrac{\begin{matrix}\mathcal{D}_1\\ z_k = E_2[t]\end{matrix}\quad(su)\dfrac{\begin{matrix}\mathcal{D}_k\\ z_{k-1} = C_k^1[z_k]\end{matrix}\quad\begin{matrix}\vdots\\ y = E_3C_j^3C_{j+1}^1\cdots C_{k-1}^1[z_{k-1}]\end{matrix}}{y = E_3C_j^3C_{j+1}^1\cdots C_k^1[z_{k-1}]}}{y = E_3C_j^3C_{j+1}^1\cdots C_k^1E_2[t]}$$

where

$$(su)\dfrac{\begin{matrix}\mathcal{D}_{j+1}\\ z_j = C_{j+1}^1[z_{j+1}]\end{matrix}\quad(su)\dfrac{\begin{matrix}\mathcal{D}\wedge\mathcal{E}\\ y_l = C_j^3[z_j]\end{matrix}\quad\begin{matrix}\mathcal{E}_1\\ y = E_3[y_l]\end{matrix}}{y = E_3C_j^3[z_j]}}{y = E_3C_j^3C_{j+1}^1[z_{j+1}]}$$

Fourth,

$$(su)\dfrac{(d)\dfrac{\begin{matrix}\mathcal{E}\\ y_l = E_4[x]\end{matrix}\quad\begin{matrix}\mathcal{E}_1\\ y_l = E_3[t]\end{matrix}}{x = t}\quad(su)\dfrac{\begin{matrix}\mathcal{D}\\ z_k = E_1[x]\end{matrix}\quad\begin{matrix}\mathcal{D}_1\\ y = E_2[z_k]\end{matrix}}{y = E_2E_1[x]}}{y = E_2E_1[t]}$$

With $\mathcal{D} \wedge \mathcal{E} \vdash y_l = C_j^3[z_j]$ and $E_3 \sim C_j^3 C_{j+1} \cdots C_k E_1$, we build

$$(su)\dfrac{\begin{matrix}\vdots\\ z_k = E_1[t]\end{matrix}\quad\begin{matrix}\mathcal{D}_1\\ y = E_2[z_k]\end{matrix}}{y = E_2E_1[t]}$$

where

$$(d)\dfrac{\begin{matrix}\mathcal{D}\wedge\mathcal{E}\\ y_l = C_j^3[z_j]\end{matrix}\quad\begin{matrix}\mathcal{E}_1\\ y_l = E_3[t]\end{matrix}}{z_j = C_{j+1}\cdots C_kE_1[t]}$$

The new deductions are in normal form. This follows from the facts that (i) both subdeductions $\mathcal{E}$ and $\mathcal{D}$ end with a (d)-rule and (ii) in the last two cases both subdeductions $\mathcal{D}_1$ and $\mathcal{E}_1$ are the rightmost subdeductions in the new deduction. □

Notice that the previous Lemma is true for any path deduction, not only for deduction whose hypotheses form an elementary cyclic set.

Lemma 5.4 *Let E be an elementary cyclic set. Let $\mathcal{D} \vdash x = C[x]$ be a cyclic deduction such that both occurrences of x are equal occurrences in E. Then $\mathcal{D}$ is not minimal among cyclic deductions from E.*

Proof. The proof is similar to the previous one. □

Lemma 5.5 *Let $\mathcal{D} \vdash x = C[x]$ be a minimal deduction from some elementary cyclic set E. Then $\mathcal{D}$ does not eliminate cyclic variables above some (d)-rule (in its auxiliary deductions).*

Proof. The deduction $\mathcal{D}$ being minimal, all marked variables are needed by Proposition 5.1. If the variable y is proper for a (t)-rule above some (d)-rule, the proper variable of the first (d)-rule below this (t)-rule is also cyclic. Also, assume that the cyclic variable y is proper for a (d)-rule:

$$(d)\ \frac{y = C_1[u] \quad y = C_2C_3[v]}{u = C_3[v]}$$

The deduction being reduced, the contexts C_1 and C_2 are non-trivial and, if C_3 is trivial then $u \not\equiv v$, as deductions are reflexivity-free. This means that the cyclic vertex $V(y)$ contains two needed $\mathcal{R}$-variables, namely w_1 such that $val(w_1) = C_1[u]$ and w_2 such that $val(w_2) = C_2C_3[v]$. As $\mathcal{R}$-variables posssess a single occurrence, $w_1 \not\equiv w_2$. But this contradicts the second part of Theorem 4.1. □

Therefore, we know the "external" structure of a minimal deduction in normal form. Let $(v_0, \ldots, v_n)$ be the sequence of needed vertices of the unique cycle c of the elementary cyclic set E, v_{i+1} being the first needed vertex above v_i along the cycle. By Proposition 2.6, for each vertex v_i there exists a variable x_i that occurs by the cycle in v_i, this variable may be chosen needed. By Theorem 4.1, this variable is the unique needed variable occurrence of $val(w_{i+1})$, w_{i+1} the unique needed $\mathcal{R}$-variable that belongs to v_{i+1}. Hence the deduction must prove $x_i = val(w_i)$ for each vertex v_i. Further, the auxiliary deductions are "out of" the cycles.

5.2. Unicity of Chains in Elementary Cyclic Sets. In this section, all variables are assumed to be needed. Let x be some variable of an elementary cyclic set E, and assume that x occurs by an edge e in $G(E)$. Remind that an occurrence of x in some non-strict right-hand side $y = C[x]$ is said to be associated to this equation iff the edge e is the last edge of the path $p = (V(y), O_C)$. For any $\mathcal{R}$-variable, there exists a unique associated equation. This fact is also true for $\mathcal{V}$-variables when we require that the variable occurs in its vertex by some edge. The idea behind the proof of this fact is quite simple: if this fails then we linearize one of these occurrences, the equations should remain cyclic. For this we need to prove the

Lemma 5.6 *Let E be an elementary cyclic set. Let x_1 and x_2 be two variables that occur by the edge e in $V(x_1) = V(x_2)$ according to two distinct equations $e_1 : y_1 = C_1[x_1]$ and $e_2 : y_2 = C_2[x_2]$. Assume that E is linearized into E' at occurrence O_{C_1} by substituting x_1' to x_1. Then, in $G(E')$ we have $V(x_1') = V(x_2)$.*

Proof. Notice first that the edge e is not cyclic by Theorem 4.1. Let $v = V(x_1) = V(x_2)$ and v' be the first needed vertex above v along e. This vertex is well-defined by (i) the

unique incident edge property for $\mathcal{V}$-free vertices and (ii) if v'' is non $\mathcal{V}$-free and unneeded, then v'' also has a unique incident edge by Proposition 2.6.

Then $V(y_1)$, $V(y_2) \in G(E) \uparrow v'$ as these two vertices are needed. Further $G(E) \uparrow v' = G(E') \uparrow v'$ as the graph above the source of e is a dag and we rename a variable whose vertex is the target of e. Hence $V(y_1)$, $V(y_2) \in G(E') \uparrow v'$ and $v' = V(y_1)/(O_{C_1}/O) = V(y_2)/(O_{C_2}/O)$ in $G(E')$ where O is the occurrence defined by the path $p = v', \ldots, v = (v', O)$. This establishes the result. □

Lemma 5.7 *Let E be an elementary cyclic set and x be some variable. If x occurs in $V(x)$ of $G(E)$ by an edge e, then there exists a unique associated equation $e' : y = C[x]$ such that e is the last edge of the path $(V(y), O_C)$. The occurrence $O(e, x)$ will denote O_C.*

Proof. This is true if the edge e is cyclic by Theorem 4.1. Assume that we have two distinct equations so that x occurs in $V(x)$ by e according to these two equations. By Lemma 5.6, we can linearize E by substituting x' to one of these occurrences so that $V(x') = V(x)$ in the new graph.

If $V(x)$ had a unique incident edge, this is also true in the new graph. Hence for every variable y in $V(x)$ from the old graph we have $y =_s z$, z occurs by e, by Lemma 2.5. Either $z \equiv x$ or not. In each case, we have $V(x) = V(y)$ in the new graph. This establishes that the two abstract graphs are equal, contradicting E an elementary cyclic set.

Otherwise, by considering an edge e' distinct from e and a chain between e and e' we also have that $V(y) = V(x)$ in the new graph for all variables $y \in V(x)$ in the old graph. Once more this gives a contradiction. □

Naturally, these occurrences being well-determined, we hope that between two co-incident edges, there will also exist essentially one chain. This is the content of the next lemmas. For this, we need to precise the notion of chain, with the same notations of Proposition 2.6:

Definition 5.1 *A chain as defined in Proposition 2.6 is an open chain. A closed chain is a triple (x, c, y) where c is an open chain, x occurs by e_0 and y occurs by e_n. We also define in an obvious way left (resp. right) closed and right (resp. left) open chains.*

By extension, a closed chain of length 0 is a pair (x, y) of variables such that $x =_s y$. A closed chain of length 1 is a triple (x, e, y) such that both x and y occur by the edge e. Such chains will be used in proving strict equations. Given two consecutive edges of a chain, the variable pairs (x_i, y_i) of a chain are unique:

Lemma 5.8 *Let E be an elementary cyclic set and v be some vertex of $G(E)$ such that e_1 and e_2 are two distinct edges both incident to v. Assume that x, y occur in v by e_1 and x', y' occur in v by e_2. Then $x =_s x'$ and $y =_s y'$ implies $x \equiv y$ and $x' \equiv y'$.*

Proof. We establish $x \equiv y$ by contradiction. Assume $x \not\equiv y$. By Lemma 5.6, if we linearize E into E' at $O(e,x)$ by substituting x'' for x, we have $V(x'') = V(y)$ in the new graph. But $y =_s y'$ implies $V(y) = V(y')$. Further $x =_s x'$ and x' still occurs by e_2 implies

$$V(x) = V(x') = V(y') = V(y) = V(x'').$$

Hence the vertices of x and x'' are equal in the new graph. This means that the graphs underlying $G(E)$ and $G(E')$ are equal, which contradicts E an elementary cyclic set. □
The argument detailed in the above proof will be frequently used in the following lemmas. As an immediate consequence, we have the following

Corollary 5.9 *Assume that in $G(E)$, E and elementary cyclic set, we have two open chains in a vertex v that share their sequence of edges, then the chains are equal.*

Definition 5.2 *Let E be a set of equations. Let v be some vertex in $G(E)$. A block of v is a set B containing at least three edges incident to v so that there exists a $\mathcal{V}$-variable x that occurs by e, for all e in B.*

Let c be a chain of v. If c contains $(e_1, \ldots, e_n)$ that forms a block, any permutation of $(e_2, \ldots, e_{n-1})$, any subsequence $(e_1, e_{i_1}, \ldots, e_{i_k}, e_n)$ also defines a chain. We are interested in minimal chains. Notice first that such a chain does not repeat any edge. Further a minimal chain does not contain any subsequence of edges defining a block. According to the equality that is to be proven, the chain will be open, closed or left-open and right-closed.

Lemma 5.10 *Let c_1 and c_2 be two minimal open chains between the edges e_1 and e_2 both incident to v. If their sets of edges are equal, then their sequences of edges also are equal.*

Proof. Let $(e_0, \ldots, e_{n+1})$ and $(e'_0, \ldots, e'_{n+1})$ be the two sequences of edges. Let i be the first index so that $e_i \neq e'_i$ and j, k be the indices such that $e'_j = e_i$ and $e_k = e'_i$. The two chains are

$$c_1 = (\ldots, x_{i-1}, y_{i-1}, e_i, x_i, \ldots, x_{k-1}, y_{k-1}, e_k, x_k, y_k, \ldots),$$
$$c_2 = (\ldots, x_{i-1}, y'_{i-1}, e'_i, x'_i, \ldots, x'_{j-1}, y'_{j-1}, e'_j, x'_j, y'_j, \ldots).$$

They share a common prefix, including the same variables, up to x_{i-1}, by Lemma 5.6. We first prove that

(i) $x_i \not\equiv x'_j$,

(ii) $y_{i-1} \not\equiv y'_{j-1}$,

(iii) $x_l \not\equiv y_{l-1}$, $l = 1, \ldots, n$,

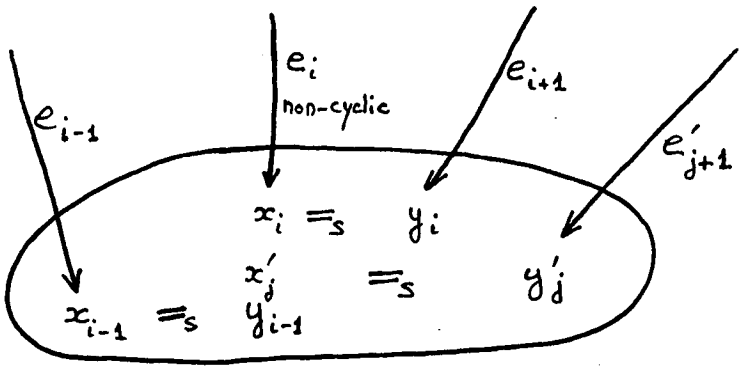

Figure 13: Structure of a vertex with two minimal chains.

Otherwise, each one of (i), (ii) implies that a chain is not minimal, e.g., $x'_j \equiv x_i$ implies that the chain

$$(\ldots, y_{i-1}, e_i, x_i, y'_j, e'_{j+1}, x'_{j+1}, \ldots)$$

does not contain the edge e'_i, while (iii) implies the existence of some block in a minimal chain.

Next, we prove that $y'_{j-1} \equiv x_i$ implies $x'_j \equiv y_{i-1}$. Otherwise, assume that $y'_{j-1} \equiv x_i$ and $y_{i-1} \not\equiv x'_j$. By (i) and (iii), we have three pairwise distinct variables, x_i, x'_j and y_{i-1}, that occur in v by e_i. By the chains c_1 and c_2, we have the strict equalities

1. $x_i =_s y_i$ that occurs by e_{i+1},
2. $x'_j =_s y'_j$ that occurs by e'_{j+1},
3. $y_{i-1} =_s x_{i-1}$ that occurs by e_{i-1}.

By the absence of repetitions in chains, we have $e_{i+1} \neq e_{i-1}$ and $e'_{j+1} \neq e_{i-1} = e'_{i-1}$. Assuming $e_{i+1} = e'_{j+1}$, as $e_i = e'_j$, we may apply Lemma 5.8 from the equalities 1 and 2. We get $x_i \equiv x'_j$ which contradicts (i). Hence the three vertices, e_{i-1}, e_{i+1} and e'_{j+1}, are pairwise distinct. We claim that, among the three variable occurrences by e_i, one can be linearized without modification of the graph. The edge e_i is not cyclic by Theorem 4.1, and we can apply Lemma 5.6 to see that this does not modify the graphs. Cf. Fig. 13: we have a vertex v with four incident edges, one among them, e_i, is associated to three distinct variables, each other one is associated to a variable strictly equal to one of the preceding ones. By Lemma 5.6, linearizing one occurrence by e_i implies that the vertex of the new variable is the same as the vertex of one of the two remaining occurrences by e_i, which in turn is our vertex v by the unmodified strict equalities.

We have proved that either $x_i \not\equiv y'_{j-1}$ or $x_i \equiv y'_{j-1}$, the last equality implying $x'_j \equiv y_{i-1}$. We conclude the proof by a case analysis.

In the former case, as $x_i \not\equiv y'_{j-1}$ and $y_{i-1} \not\equiv y'_{j-1}$ by (ii), the chain c_1 still exists when we linearize the occurrence of y'_{j-1} by e_i.

In the latter case, we get a smaller chain c_3 without e_i: $c3 = (\ldots, x_{i-1}, y_j, e'_{j+1}, \ldots)$ as $x_{i-1} =_s y_{i-1} \equiv x'_j =_s y'_j$. This concludes the proof. □

Corollary 5.11 *Two minimal chains c_1 and c_2 with the same sets of edges are equal.*

Lemma 5.12 *Let c_1 and c_2 be two minimal chains between e_1 and e_2 incident to v. Then they have the same edges.*

Proof. Assume that c_1 and c_2 differ by say $e \in c_1$, $e \notin c_2$. Both c_1 and c_2 being minimal, y_i and x_{i+1} that occur by e in v are distinct. Hence e is not cyclic, we can linearize E at the occurrence of y_i or x_{i+1} by Lemmas 5.6 and 5.7 without modifying the graph. □

Proposition 5.13 *Let E be an elementary cyclic set and v be a shared vertex of $G(E)$. For all pairs (e_1, e_2) of distinct edges incident to v there exists a minimum open chain between e_1 and e_2.*

Proof. Consequence of Lemma 5.10 and Corollaries 5.9 and 5.11. □

Of course, we establish the same results for other kinds of chains.

Lemma 5.14 *Let E be an elementary cyclic set and x, y be two distinct variables. If $x =_s y$ there exist a minimum (t)-deduction of this equality.*

Proof. A reduced (t)-deduction is a sequence of variables x_i, $i = 0, \ldots, n$ with $x_0 \equiv x$, $x_n \equiv y$ and $x_i = x_{i+1} \in S$. Assume that there exists two distinct sequences (x_i) and (x'_i). Let j be the first index such that $x_j \not\equiv x'_j$. Then as $x =_s x_j$, $x =_s x'_j$, $y =_s x_j$, $y =_s x'_j$ we can remove, say the equation $x_{j-1} = x_j$ without modifying the graph, contradiction. □

Proposition 5.15 *Let E be some elementary cyclic set and x, y be two variables with $V(x) = V(y)$. Then there exists a minimum closed chain between x and y.*

Proof. We first establish the existence of such chains. If $x =_s y$, there exists a (t)-deduction of this equation by definition of $=_s$. Otherwise $v = V(x) = V(y)$ is not a root. By Lemma 2.5, there exists a and b such that $a =_s x$, a occurs by some edge e_1 in v, $b =_s y$, b occurs by some edge e_2 in v. By Proposition 5.13, there exists a minimal chain between e_1 and e_2. This establishes the existence of a chain proving $a = b$, hence $x = y$.

We now establish the unicity. Assume $x =_s y$. By Lemma 5.14, there exists a minimum (t)-deduction of $x =_s y$. Otherwise, we firstly assume that in minimal chains only one edge is involved and that (1) a and b occur by e, with $a \neq_s b$, $x =_s a$ and $y =_s b$, (2) a' and b' occur by e', with $a' \neq_s b'$, $x =_s a'$ and $y =_s b'$. If $e = e'$, but say $a \not\equiv a'$, we can

linearize e.g. a by e, as $a =_s a'$. But $e \neq e'$ is impossible by Lemma 5.8, as $a =_s a'$ and $b =_s b'$.

Secondly, assume that $n+1$ edges, $n > 0$, are involved in minimal chains and that we have two closed minimal distinct chains proving the equality $x = y$:

$$c = (a, e_0, x_0, y_0, \ldots, x_{n-1}, y_{n-1}, e_n, b),$$

$$c' = (a', e'_0, x'_0, y'_0, \ldots, x'_{n-1}, y'_{n-1}, e'_n, b').$$

Then if $e_0 = e'_0$ and $e_n = e'_n$, by Proposition 5.13, we have $a \not\equiv a'$ or $b \not\equiv b'$, say $a \not\equiv a'$. But $a =_s a'$ and we may linearize say the occurrence of a by e_0. Otherwise $e_0 \neq e'_0$. Then we have $a =_s a'$, $x_0 \neq_s a$ and $x'_0 \neq_s a'$ by minimality of the chains. As a, x_0 occur by e_0 and a', x'_0 occur by e'_0, once more we may linearize. □

Proposition 5.16 *Let E be an elementary cyclic set. Let $x \in v$ and e being incident to v. Then there exists a minimum left-closed and right-open chain that starts with x and ends with y, for some y that occurs in v by e.*

Proof. Existence: by Proposition 2.6, v being non $\mathcal{V}$-free, there exists at least one variable y that occurs by e in v. For every such y there exists a minimum closed chain that proves $x = y$ by Proposition 5.15.

Unicity: if x occurs by e the unicity follows from Lemma 5.7. Next, assume firstly that x is strictly equal to some variable that occurs by e. If there exists two such variables y and y', then $x \not\equiv y$, $x \not\equiv y'$ and $y \not\equiv y'$ imply that we may delete some equation involved in the (t)-deduction of, e.g., $x =_s y$, contradiction. Secondly, assume that we have two minimal left-closed right-open distinct chains:

$$c = (z, e_0, x_0, y_0, \ldots, x_{n-1}, y_{n-1}, e_n),$$

$$c' = (z', e'_0, x'_0, y'_0, \ldots, x'_{n-1}, y'_{n-1}, e'_n).$$

If $e_0 = e'_0$ then the open chains are equal by Proposition 5.13. We are in the previous case: z, z' occur by e, both are strictly equal to x, we may delete some equation of E involved in e.g. $x =_s z$. Finally, if $e_0 \neq e_0$, then we may also delete the same strict equation, the graph still are equal by Lemma 5.6 and the existence in the new graph of the two open chains associated to c and c'. □

These three propositions detail the three cases where we will need the unicity property in the next section. In addition we have

Proposition 5.17 *Let v be some cyclic needed vertex in $G(E)$, E some elementary cyclic set. There exists a maximum open chain in v.*

Proof. Direct consequence of Corollary 5.11, as we do not have any block in a needed cyclic vertex. Hence all open chains are minimal and there trivially exists at least one maximal chain. □

5.3. Unicity and Sequentiality of the Minimal Deduction. We are now in position to define a proof-search procedure. Assume that at some step in the execution of this algorithm we visit some vertex v from which we have to prove some equation between terms rooted in this vertex. If this vertex is non-root, this equation will most probably depend on the graph $G \uparrow v$. The knowledge of the equation and the existence of minimum chains imply that we can search this subproof through the minimum chains. We formalize this idea. A cyclic variable x of the vertex v is the proper variable of v iff x is the (unique) variable that occurs in v by the edge incident to v belonging to the cycle.

Theorem 5.18 *Let x be some proper variable of the elementary cyclic set E. There exists a minimum deduction $\mathcal{D} \vdash x = C[x]$ of the cyclic equation associated to x. The minimum deductions for other proper variables are obtained from $\mathcal{D}$ by a cyclic permutation of the auxiliary deductions of $\mathcal{D}$.*

Proof. We establish the result by giving a deterministic algorithm that searches such a deduction. The inference rules are denoted by function symbols of arity 2: SU for the substitution, T for the transitivity and D for the simplification rule. The algorithm includes a main body and three mutually recursive functions. The first one *Connect* returns a deduction of $x_{i+1} = C_i[x_i]$ where x_{i+1} (resp. x_i) is the proper variable of the cyclic needed vertex v_{i+1} (resp. v_i). The function *Chain* takes an open chain from x_0 to y_{n-1} (with the usual notations) and returns a deduction of $x_0 = y_{n-1}$. Finally, the function *Edge* takes an edge and two (distinct) variables that occur by this edge (either $\mathcal{R}$- or $\mathcal{V}$-variables) and returns a deduction of their equality. The intuition behind this last function is: the equations associated to these variables are unique. Also, we have three distinct cases according to the relative position of the paths defined by these two equations: they are equal, one is suffix of the other, or they diverge at some vertex. One of the three Propositions 5.13, 5.15 or 5.16 applies to each case.

CYCLE DEDUCTION

Input: an elementary cyclic set E, its graph $G(E)$,
some needed cyclic vertex v_0;
Let $v_0, \ldots, v_n$ be the sequence of needed cyclic vertices of $G(E)$, v_{i+1} the first needed cyclic vertex above v_i along the cycle, with $v_{n+1} = v_0$;
For $i = 0, \ldots, n$ Let $T_i = Connect(v_i, v_{i+1})$;
Return $(S(T_0, S(\cdots S(T_{n-1}, T_n)\cdots)))$.

Procedure $Connect(v_1, v_2)$

If v_2 possesses a unique incident edge
Then Let $e : x = C[y]$ be the unique non-strict equation of v_2;
Let z be the variable of v_2 that possesses an occurrence
by the cycle;
Let $\mathcal{D}$ be the minimum (t)-deduction of $z =_s x$;
Return($T(\mathcal{D}, e)$ or e if $\mathcal{D}$ is void);
Else Let $c = (e_0, x_0, y_0, e_1, \ldots, e_n, x_{n-1}, y_{n-1}, e_n)$ be the maximum chain of v_2
such that w, the unique needed $\mathcal{R}$-variable of v_2 occurs by e_n,
and e_0 is the cyclic edge incident to v_2;
Let $\mathcal{D}_1 = Chain(v_2, c)$;
Let $\mathcal{D}_2 = Edge(y_{n-1}, e, w)$;
Return($T(\mathcal{D}_1, \mathcal{D}_2)$).

Procedure $Chain(e_0, x_0, y_0, e_1, \ldots, e_n, x_{n-1}, y_{n-1}, e_n)$

For $i = 0, \ldots, n-1$ Let $\mathcal{D}_i^1$ be the minimum (t)-deduction of $x_i =_s y_i$;
For $i = 0, \ldots, n-1$ Let $\mathcal{D}_i^2 = Edge(v, y_i, e_{i+1}, x_{i+1})$;
For $i = 0, \ldots, n-1$ If $\mathcal{D}_i^1$ is void Then $\mathcal{D}_i^3 = \mathcal{D}_i^2$ Else $\mathcal{D}_i^3 = T(\mathcal{D}_i^1, \mathcal{D}_i^2)$;
Let $\mathcal{D} = \mathcal{D}_1^3$;
For $i = 2, \ldots, n-1$ Let $\mathcal{D} = T(\mathcal{D}, \mathcal{D}_i^3)$;
Return($T(\mathcal{D}, \mathcal{D}_{n-1}^1)$ or $\mathcal{D}$ if $\mathcal{D}_{n-1}^1$ is void).

Procedure $Edge(x, e, y)$

Let $e_1 : x = C[x]$ and $e_2 : y = D[y]$ be associated to $O(x, e)$ and $O(y, e)$;
Let p be the maximal common suffix of $p_1 = (V(x), O_C)$ and $p_2 = (V(y), O_D)$;
If $p = p_1 = p_2$
Then If $x \equiv y$
Then Return($D(e_1, e_2)$);
Else Let $c = (y, e_0, x_0, \ldots, y_{n-1}, e_n, x)$ be
the minimum closed chain between y and x;
Let $\mathcal{D}_1 = Edge(y, e_0, x_0)$;
Let $\mathcal{D}_2 = Chain(e_0, x_0, \ldots, y_{n-1}, e_n)$;
Let $\mathcal{D}_3 = Edge(y_{n-1}, e_n, y)$;
Let $Tr = T(T(\mathcal{D}_1, \mathcal{D}_2), \mathcal{D}_3)$;
Return($D(T(Tr, e_1), e_2)$);
If $p = p_1$
Then Let e_0 be the edge of p_2 incident to $V(x)$;
Let w be the $\mathcal{R}$-variable associated to $D[y]$
that occurs by e_0 in $V(x)$;
Let $c = (e_0, x_0, \ldots, y_{n-1}, e_n, x)$ be

the minimum left-open right-closed chain between e_0 and x;
Let $\mathcal{D}_1 = Chain(e_0, x_0, \ldots, y_{n-1}, e_n)$;
Let $\mathcal{D}_2 = Edge(y_{n-1}, e_n, x)$;
Let $\mathcal{D}_3 = Edge(x_0, e_0, w)$;
Let $Tr = T(\mathcal{D}_1, \mathcal{D}_2)$;
Return$(D(T(Tr, e_1), \mathcal{D}_3))$;

If $p = p_2$
Then Let e_n be the edge of p_1 incident to $V(y)$;
Let w be the $\mathcal{R}$-variable associated to $C[x]$
that occurs by e_n in $V(y)$;
Let $c = (y, e_0, x_0, \ldots, y_{n-1}, e_n)$ be
the minimum left-closed right-open chain between y and e_n;
Let $\mathcal{D}_1 = Edge(y, e_0, x_0)$;
Let $\mathcal{D}_2 = Chain(e_0, x_0, \ldots, y_{n-1}, e_n)$;
Let $\mathcal{D}_3 = Edge(y_{n-1}, e_n, w)$;
Let $Tr = T(\mathcal{D}_1, \mathcal{D}_2)$;
Return$(D(T(Tr, \mathcal{D}_3), e_2))$;

Else Let v' be the source of the path p;
Let e, e' be the two edges of the paths p_1 and p_2
that occur in v';
Let w_0, w_1 be the two corresponding $\mathcal{R}$-variables;
Let $c = (e_0, x_0, \ldots, y_{n-1}, e_n)$ be the minimum open chain
between e' and e;
Let $\mathcal{D}_1 = Edge(x_0, e_0, w_1)$;
Let $\mathcal{D}_2 = Chain(e_0, x_0, \ldots, y_{n-1}, e_n)$;
Let $\mathcal{D}_3 = Edge(y_{n-1}, e_n, w_0)$;
Return$(D(T(\mathcal{D}_2, \mathcal{D}_3), \mathcal{D}_1))$.

The correction of the algorithm follows from the Lemmas in section 5, i.e. the algorithm terminates and computes a cyclic deduction. The computed deduction is minimum by Propositions 5.13, 5.15, 5.16 and 5.17, i.e., at each recursive call, the minimum amount of computation is performed in order to prove some equality. □

Let us see an example. It contains a unique cycle, which is primitive (cf. Fig. 14):

$x = f(l, a')$ $\quad x = f(m, b')$ $\quad x = f(f(c', f(f(b, d'), e')), f')$
$y = f(m, g')$ $\quad y = f(n, h')$ $\quad y = f(f(i', f(j', c)), k')$
$z = f(o, l')$ $\quad z = f(n, m')$
$o = f(b, n')$ $\quad l = f(f(o', c), p')$ $\quad n = f(q', f(a, a))$

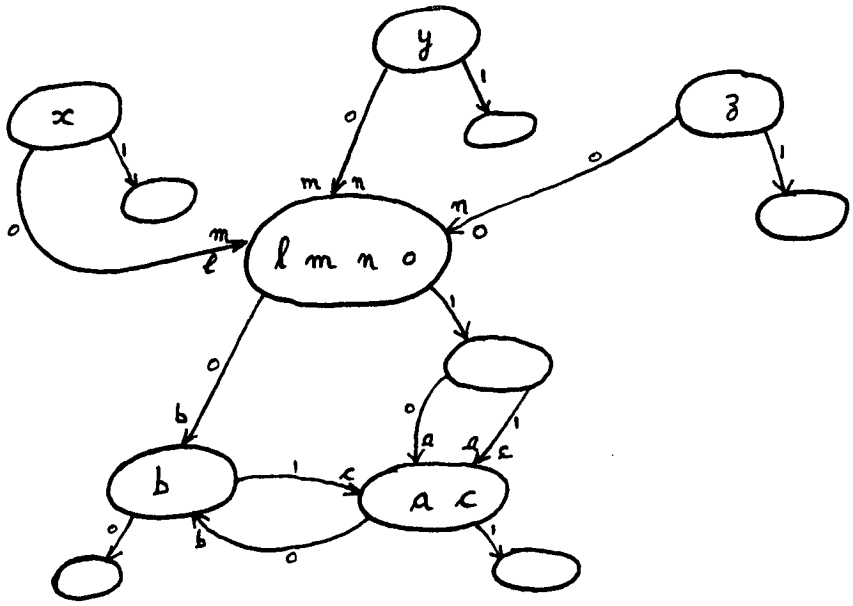

Figure 14: Graph $G(\mathcal{E}_5)$

We are now able to decipher this example $\mathcal{E}_5$. The needed variables are the unprimed ones. The two proper variables of the cyclic vertices are b and c. The associated equations are $x = f(f(c', f(f(b, d'), e')), f')$ and $l = f(f(o', c), p')$. The occurrences of the needed variables are displayed on the graph $G(\mathcal{E}_5)$. The minimum cyclic deductions for b and c are as follows. The first auxiliary deduction $\mathcal{D}_1$ is:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x=f(l,a') \quad x=f(m,b')}{l=m} \quad \dfrac{y=f(m,g') \quad y=f(n,h')}{m=n}}{l=n} \quad \dfrac{z=f(n,m') \quad z=f(o,l')}{n=o}}{l=o} \quad o=f(b,n')}{l=f(b,n')} \quad l=f(f(o',c),p')}{b=f(o',c)}$$

Let $\mathcal{D}_2'$ and $\mathcal{D}_2''$ be the two deductions:

$$\dfrac{\dfrac{\dfrac{y=f(n,h') \quad y=f(f(i',f(j',c)),k')}{n=f(i',f(j',c))} \quad n=f(q',f(a,a))}{c=a} \quad \dfrac{\dfrac{y=f(m,g') \quad y=f(n,h')}{m=n} \quad n=f(q',f(a,a))}{m=f(q',f(a,a))} \quad \dfrac{x=f(m,b') \quad x=f(f(c',f(f(b,d'),e')),f')}{m=f(c',f(f(b,d'),e'))}}{a=f(b,d')}$$

The second auxiliary deduction is $\mathcal{D}_2$:

$$\dfrac{\begin{matrix}\mathcal{D}_2' & \mathcal{D}_2'' \\ c=a & a=f(b,d')\end{matrix}}{c=f(b,d')}$$

The two minimum deductions are :

$$(su)\ \frac{\begin{array}{cc}\mathcal{D}_2 & \mathcal{D}_1\\ c = f(b,d') & b = f(o',c)\end{array}}{b = f(o', f(b,d'))} \qquad (su)\ \frac{\begin{array}{cc}\mathcal{D}_1 & \mathcal{D}_2\\ b = f(o',c) & c = f(b,d')\end{array}}{c = f(f(o',c),d')}$$

Notice that the two equations above are the rightmost hypotheses in the auxiliary deductions. To conclude, we mention that the analysis of occur-checks also applies to homogeneity tests, for unification over an arbitrary signature Σ. The notion of elementary cyclic set must be replaced by the definition of elementary non-homogeneous set, the existence of a cycle being replaced by the existence of a failure by homogeneity. We assume that the input equations still are of the form $x = t$. The inference rules for simplification require that the head symbols of the members of the premiss are equal. In other words we can deduce $f(\cdots) = g(\cdots)$, $f \neq g \in \Sigma$.

Theorem 5.19 *Let $\mathcal{E}$ be a unification problem on some signature Σ. Then there exists a finite basis of elementary non-homogeneous sets embedded in $\mathcal{E}$. For each element of the basis, there exists a minimum, (su)-free, equational deduction of a non-homogeneous equation.*

Sketch of Proof. The proof of the unicity follows the lines of the cycle unicity proof in section 3. The syntactical analysis of deductions is valid (cf. especially the proof of Lemma 2.8). The semantical analysis remains valid, essentially because the Lemmas in section 5.1 are true for proofs whose hypotheses do not form an elementary set. The algorithm is a subcase of the `Cycle Deduction` procedure. □

Let $\mathcal{E}$ be a set of equations. An exhaustive search gives all the elementary cyclic sets imbedded in $\mathcal{E}$. Call the set of their canonical deductions $\mathcal{B}(\mathcal{E})$. Now trivially, $\mathcal{E}$ is unifiable iff $\mathcal{B}(\mathcal{E})$ is empty. But we have something more: the set $\mathcal{B}(\mathcal{E})$ encodes the minimal amount of computation that has to be performed in order to prove that $\mathcal{E}$ is ununifiable. Formalizing this argument would lead us too far. We conjecture that some results about the complexity of subclasses of the unification problem [7,33] follow easily from the description of their base $\{\mathcal{B}(\mathcal{E}) \mid \mathcal{E} \in C\}$. For example, if the number of repeated variables is bounded, the height of the deductions in the base $\mathcal{B}(\mathcal{E})$ is bounded uniformly on the class. Hence an efficient parallel algorithm can be designed, showing that this class is in NC while unification is P-complete. Finally, for Type Inference, some quite involved examples show that we must consider the second-order lifting of the set $\mathcal{C}(\mathcal{E})$, protecting $\mathcal{B}(\mathcal{E})$ is not sufficient, see especially the example $(\lambda x.x\ x)\ (\lambda x, y.x\ y\ x)$ in the introduction.

Acknowledgements. I wish to thank Dana Scott and Gérard Huet for a pleasant sabbatical year that gave its impetus to the present research, Jean-Yves Girard for his support during the final efforts and the referees for their patience and help.

References

[1] ANDREWS P.B. "Resolution in Type Theory." *J. of Symb. Logic* **36** (1971) 414–432.

[2] BERGE C. *Graphes.* Gauthier-Villars (1983, 3rd ed.).

[3] BARENDREGT H.K. *The Lambda Calculus: Its Syntax and Semantics.* North-Holland (1983, 3rd ed.).

[4] BERRY G., CURIEN P.L. "Sequential Algorithms on Concrete Data Structures." *Theo. Comp. Sci.* **20** (1982) 265–321.

[5] COQUAND TH. AND HUET G. "The Calculus of Constructions." *Information and Computation* **76**, 2/3 (1988) 95–120.

[6] COURCELLE B. "Fundamental Properties of Infinite Trees." *Theo. Comp. Sci.* **25** (1983) 95–169.

[7] COURCELLE B., KAHN G. AND VUILLEMIN J. "Algorithmes d'équivalence et de réduction à des expressions minimales dans une classe d'équations récursives simples." Rapport INRIA 37, (1973).

[8] DWORK C., KANELLAKIS P. AND MITCHELL J. "On the Sequential Nature of Unification." *J. of Logic Programming* **1**(1), 35–50.

[9] GALLIER J.H. *Logic for Computer Science: Foundations of Automated Reasoning.* Harper & Row, New-York (1986).

[10] GENTZEN G. *The Collected Papers of Gerhard Gentzen.* Ed. E. Szabo, North-Holland, Amsterdam (1969).

[11] GIANNINI P., RONCHI DELLA ROCCA S. "Characterization of Typings in Polymorphic Type Discipline." Proc. 3rd LICS (1988) 61–70.

[12] GIRARD J.Y. *Proof Theory and Logical Complexity.* Studies in Proof Theory 1, Bibliopolis, Napoli (1988).

[13] GIRARD J.Y. *Interprétation fonctionnelle et élimination des coupures dans l'arithmétique d'ordre supérieur.* Thèse d'Etat, Université Paris VII (1972).

[14] GIRARD J.Y. "Linear Logic." *Theo. Comp. Sci.* **50** (1987) 1–102.

[15] GOLDFARB W. "The Undecidability of the Second-Order Unification Problem." *Theo. Comp. Sci.* **13**,2 (1981) 225–230.

[16] HENKIN L. "The Logic of Equality." *The Amer. Math. Monthly* **1977**, 597–612.

[17] HERBRAND J. *Sur la Théorie de la Démonstration.* in: *Logical Writings*, W. Goldfarb (ed.) Cambridge (1971).

[18] HOWARD W. "The Formulas-as-Types Notion of Construction." In *To H.B. Curry: Essays on Combinatory Logic, Lambda-Calculus and Formalism.* Seldin J.P. and Hindley J.R. (Eds.) Academic Press (1980) 479–490.

[19] HUET G. "Confluent Reductions: Abstract Properties and Application to Term Rewriting Systems." *JACM* **274** (1980) 797–821.

[20] HUET G. "A Unification Algorithm for Typed λ-calculus." *Theo. Comp. Sci.* **1** (1975) 27–57.

[21] HUET G. *Résolution d'équations dans les langages d'ordre 1,2,...,ω.* Thèse d'Etat, Université Paris VII (1976).

[22] KREISEL G. AND TAIT W. "Finite Definability of Number Theoretic Functions and Parametric Completeness of Equational Calculi." *Z. Math. Logik Grundlagen Math.* **7** (1961) 28–38.

[23] LE CHENADEC PH. *Canonical Forms in Finitely Presented Algebras.* Research Notes in Theoretical Computer Science, Pitman–Wiley (1986).

[24] LE CHENADEC PH. "The Finite Automaton of an Elementary Cyclic Set." INRIA Research Report 824, April 1988.

[25] LEWIS H.R. AND STATMAN R. "Unifiability is Complete for co-NLogSpace." *Inf. Proc. Letters* **15**,5 (1982) 220–222.

[26] MARTELLI A., MONTANARI U. "An Efficient Unification Algorithm." *ACM Toplas* 4,2 (1982) 258–282.

[27] PATERSON M.S., WEGMAN M.N. "Linear Unification." *JCSS* **16** (1978) 158–167.

[28] PRAWITZ D. *Natural Deduction.* Almqist and Wiskell, Stockolm (1965).

[29] PRELLER A., LAFAYE DE MICHEAUX N. "Intensional Equality in Categories with Structure and Coherence Problems." *Zeitschr. f. math. Logik und Grundlagen d. Math.* **34** (1988) 421–432.

[30] REYNOLDS J.C. "Towards a Theory of Type Structure." Paris Colloq. on Programming (1974) 408–425.

[31] ROBINSON J.A. "A Machine Oriented Logic Based on the Resolution Principle." *JACM* **12**,1 (1965) 23–41

[32] SHOENFIELD J.R. *Introduction to Mathematical Logic.* Addison-Wesley (1967).

[33] STATMAN R. "Solving Functional Equations at Higher Types; Some Examples and Some Theorems." *Notre-Dame J. of Formal Logic* **27**,1 (1986) 66–74.

[34] VITTER J.S. AND SIMONS R.A. "New Classes for Parallel Complexity: A study of Unification and Other Complete Problems for P." *IEEE Trans. on Computers* **c-35**,5 (1986) 403–418.

[35] WHITEHEAD G.W. *Elements of Homotopy Theory.* G.T.M. 61 (1978) Springer-Verlag.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique